P-114

UTEC ME 87-034

Final Report

to the

National Aeronautics and Space Administration

Grant NAG-1-283

ACOUSTIC PROPAGATION IN A THERMALLY STRATIFIED ATMOSPHERE

W. K. Van Moorhem

Mechanical and Industrial Engineering Department
University of Utah
Salt Lake City, Utah 84112

September, 1987

ABSTRACT

Acoustic propagation in an atmosphere with a specific form of a temperature profile has been investigated by analytical means. The temperature profile used is representative of an actual atmospheric profile and contains three free parameters. Both lapse and inversion cases have been considered. Although ray solution have been considered the primary emphasis has been on solutions of the acoustic wave equation with point source where the sound speed varies with height above the ground corresponding to the assumed temperature profile. The method used to obtain the solution of the wave equation is based on Hankel transformation of the wave equation, approximate solution of the transformed equation for wavelength small compared to the scale of the temperature (or sound speed) profile, and approximate or numerical inversion of the Hankel transformed solution. The solution display the characteristics found in experimental data but extensive comparison between the models and experimental data has not been carried out.

TABLE OF CONTENTS

1.0 INTRODUCTION

Although the propagation of acoustic signals through the atmosphere has been studied for many years, and most atmospheric effects are understood in the qualitative sense, quantitative modeling of most of these effects has become an area of interest only recently. The dominant effect occurring in atmospheric propagation is the spreading of acoustic energy associated with a wave propagating in three dimensions over an ever increasing area, the well known spherical spreading effect, which occurs in an isothermal, unbounded atmosphere. In addition to this, acoustic waves are absorbed by the atmosphere, reflected and absorbed at the ground surface, scattered by turbulence, and refracted by both wind and temperature gradients.

This report summarizes a project to develop models for the propagation of acoustic signals from a point source above a finite impedance ground surface in the presence of temperature gradients in the atmosphere. The situation of interest is the case of sound from a source located within a few meters of the ground propagating to a receiver located within a few meters of the ground through the temperature gradient that commonly occurs just above the ground surface. Best[1], Gieger[2], and Reynolds[3] all discuss the temperature gradient in this region. Within one to two meters of the ground the temperature generally goes through a diurnal cycle with a lapse condition, temperature decreasing with height, occurring in the afternoon and an inversion condition, temperature increasing with height, at night, see Figure 1.1. Shortly after sunrise and sunset the atmosphere goes through a nearly isothermal period when the transition from lapse to inversion or inversion to lapse condition is under way. This simple picture of the very complex atmospheric dynamics near the ground can be upset by significant winds which increase the mixing near the ground surface and tend to lead to a more isothermal situation, or to an overcast which can prevent a strong lapse condition from developing by blocking the insolation or prevent an inversion from occurring by blocking the radiation from the ground to the night sky.

References 1,2 and 3 all discuss the classic logarithmic temperature profile given by

$$T = T_o \ln\left(\frac{z}{z_o}\right) \tag{1.1}$$

which is based on empirical results. This result, although fitting the experimental data well, certainly is not reasonable either for heights very near the ground or very far above the ground. In addition, logarithmic functions are generally more difficult to deal with in an analysis then are algebraic functions. For this latter reason the profile used in this study is

$$T = T_\infty + \frac{\Delta T}{1 + \alpha z} \tag{1.2}$$

This form of the temperature profile is shown in Figure 1.2 along with some temperature data obtained by Butterworth[4]. The agreement between the data and the assumed function fitted to this data is excellent. Also as compared to (1.1) the physical meaning of the parameters in (1.2), T_∞, ΔT, and α, are clear. The assumed temperature profile asymptotically approaches the temperature T_∞ high above the ground. At the ground the temperature is $T_\infty + \Delta T$, thus the change in temperature between the ground and far about the ground is ΔT. The derivative of temperature with respect to height evaluated at the ground surface is $-\alpha\,\Delta T$. Thus $1/\alpha$ is the scale over which the temperature change ΔT occurs. For example at a height $z = 1/\alpha$ one-half of the total temperature change ΔT has occurred. The temperature profile of (1.2) can be used to represent either a lapse or inversion condition. For a lapse condition the parameter ΔT is positive and for an inversion it is negative.

The equation governing the wave motion is the simple acoustic wave equation with a sound speed varying with height and with a point source term,

$$\frac{1}{a(z)^2}\frac{\partial^2 p}{\partial t^2} - \nabla^2 p = \frac{q}{\pi r} e^{i\omega t}\,\delta(r)\,\delta(z-s) \tag{1.3}$$

At the ground surface, $z = 0$, a normal impedance boundary condition

$$p = \frac{Z}{i\rho\omega}\frac{\partial p}{\partial z} \tag{1.4}$$

is assumed. High above the ground, $z \rightarrow \infty$, only outgoing waves are permitted, a radiation condition. At the source height, $z = s$, the pressure field is to be continuous, and to satisfy the conditions implied by (1.3).

Section 2 contains a discussion of the acoustic rays that characterize both the lapse and inversion cases for the assumed temperature gradient, (1.2). This both yields a quantitative understanding of the propagation phenomena, and plays an integral part in understanding the modeling that follows. The model for the lapse condition is developed in Section 3 and the inversion model is described in Section 4.

Section 5 contains a discussion of the conclusions developed during the project.

2.0 ACOUSTIC RAYS

Acoustic ray tracing is a relatively simple procedure for an axisymmetric case which yields a great deal of qualitative information about a given propagation situation. Only a brief discussion is given here. More detail is given in [5]. For an horizontally stratified atmosphere the acoustic rays may be determined from an acoustic form of Snell's law

$$\frac{\text{Cos } \theta(z)}{a(z)} = \frac{\text{Cos } \theta(s)}{a(s)} \tag{2.1}$$

where the source is located at the height s. Here θ(z) is the angle between a ray and the horizontal at the height z, see Figure 2.1. Thus the right hand side of (2.1) is a constant for a ray emitted from the source at an initial angle θ(s). Using (1.2) to obtain

$$a^2(z) = a_\infty^2 \left(1 + \frac{\Delta T}{T_\infty} \frac{1}{(1 + \alpha z)}\right) \tag{2.2}$$

which describes the sound speed as a function of height, the slope of a ray initially emitted from the source at an angle θ(s) is determined from (2.1) to be given by

$$\frac{dz}{dr} = \pm \sqrt{\frac{A z + B}{C z + D}} \tag{2.3}$$

where

$$A = \alpha \left[1 + \alpha s + \frac{\Delta T}{T_\infty} - (1 + \alpha s)\, \mathrm{Cos}^2\theta(s) \right]$$

(2.4)

$$B = \left[1 + \alpha s + \frac{\Delta T}{T_\infty} - (1 + \alpha s) (1 + \frac{\Delta T}{T_\infty})\, \mathrm{Cos}^2\theta(s) \right]$$

(2.5)

$$C = \alpha (1 + \alpha s)\, \mathrm{Cos}^2\theta(s)$$

(2.6)

and

$$D = (1 + \frac{\Delta T}{T_\infty}) (1 + \alpha s)\, \mathrm{Cos}^2\theta(s)$$

(2.7)

Equation (2.3) can be integrated to obtain the ray paths. Different results are obtained in the lapse and inversion cases and these will be considered separately in the next two sections.

2.1 Lapse Case

In the lapse case the quantity A is positive and integrating (2.3) to obtain the rays yields four different cases. For rays going upward from the source (using the positive sign in (2.3))

$$r = F(z, \theta(s)) - F(s, \theta(s)) \tag{2.6}$$

for rays going downward initially from the source (using the negative sign)

$$r = F(s, \theta(s)) - F(z, \theta(s)) \tag{2.7}$$

for rays that were initially going downward but have been reflected upward at the ground (using the positive sign and (2.7))

$$r = F(z, \theta(s)) + F(s, \theta(s)) - 2\,F(0, \theta(s)) \tag{2.8}$$

and for rays that initially were going downward and were refracted upward before reaching the ground (again using the positive sign and (2.7))

$$r = F(z, \theta(s)) + F(s, \theta(s)) \tag{2.9}$$

The function F(z, θ(s)) is given by

$$F(z, \theta(s)) = \frac{\sqrt{(A\,z + B)(C\,z + D)}}{A} + \frac{E}{2\,A\sqrt{A\,C}} \ln\left(\frac{1+\phi}{1-\phi}\right) \tag{2.10}$$

where

$$E = AD - BC$$
$$= \alpha\left(1 + \alpha s + \frac{\Delta T}{T_\infty}\right)\left(\frac{\Delta T}{T_\infty}\right)(1 + \alpha s)\, \mathrm{Cos}^2\theta(s) \qquad (2.11)$$

and

$$\phi = \sqrt{\frac{C(Az+B)}{|A|(Cz+D)}} \qquad (2.12)$$

The absolute value of A in (2.12) is immaterial in the lapse case where A is always positive but significant in the case of an inversion where A may change sign.

The rays are identified by the parameter θ(s), the initial angle at which the ray leaves the source. Thus a ray initially propagating downward and identified by a particular value of θ(s) will either be reflected upward at the ground or refracted upward at a turning point. In either case the reflected or reflected ray will be identified by the same value of θ(s) as the initial ray. Figure 2.2 is an example of the rays calculated from (2.6) to (2.9).

Setting θ(z) = 0 in (2.3) yields an expression for the height at which an initially downward propagating ray becomes horizontal, the turning point, as

$$z_{tp} = -\frac{B}{A}$$

(2.13)

Solving this expression for Cos θ(s) yields

$$\text{Cos } \theta(s) = \sqrt{\frac{(1 + \alpha z_{tp})(1 + \alpha s + \frac{\Delta T}{T_\infty})}{(1 + \alpha s)(1 + \alpha z_{tp} + \frac{\Delta T}{T_\infty})}}$$

(2.14)

which identifies the ray having a turning point at a height z_{tp}. The ray that grazes the ground and is the boundary between reflected and refracted rays can be found by setting $z_{tp} = 0$ in (2.14). The ray that divides the initially upward and downward propagating rays is identified by $\theta(s) = 0$.

For the ray that grazes the ground, and is identified by the value of θ(s) defined by (2.14) with $z_{tp} = 0$, the function $F(0, \theta(s)) = 0$ and thus this ray is defined by either (2.8) or (2.9). Similarly for $\theta(s) = 0$ the ray can either be obtained from (2.6) or (2.9) since on this ray $F(s, 0) = 0$. At a turning point $F(z_{tp}, \theta(s)) = 0$ when θ(s) is given by (2.14).

The shadow boundary is more difficult to locate. It is bounded by refracted rays (2.9) and at a fixed height z is the maximum possible value of r for rays with turning points below that height. Thus the ray tangent to the shadow boundary or caustic at the height z is identified by solving the equation

$$\frac{\partial r}{\partial \theta(s)} = \frac{\partial F(z, \theta(s))}{\partial \theta(s)} + \frac{\partial F(s, \theta(s))}{\partial \theta(s)} = 0$$

(2.15)

for θ(s) and then using that value in (2.9) to determine the location. The derivative in (2.15) can be obtained from

$$\frac{\partial F}{\partial \gamma} = \frac{\{(1+\alpha z)(1-\gamma^2)-3\gamma^2\}\sqrt{1+\alpha z+\frac{\Delta T}{T_\infty}}}{\alpha(1-\gamma^2)^2\sqrt{1+\alpha z-\gamma^2(1+\alpha z+\frac{\Delta T}{T_\infty})}} + \frac{\frac{\Delta T}{T_\infty}[1+2\gamma^2]}{2[1-\gamma^2]^{\frac{5}{2}}}\ln\left(\frac{1+\phi}{1-\phi}\right)$$

(2.16)

where

$$\gamma = \sqrt{\frac{1+\alpha z}{1+\alpha z+\frac{\Delta T}{T_\infty}}}\,\text{Cos}\,\theta(s)$$

(2.17)

Due to the complexity of this expression an analytic solution is not possible and either a numerical solution of (2.15) must be obtained or the approximate relations given in [5] must be used.

2.2 Inversion Case

As in the lapse case several different types of rays occur in the inversion case. Integrating (2.3) with $\Delta T/T_\infty$ negative also yields three different forms for the function which determines the rays depending upon whether the quantity A given in (2.4) is positive, negative or zero. The meaning of these three cases is discussed below. For rays that are initially angled upward (using the positive sign in (2.3))

$$r = F_i(z, \theta(s)) - F_i(s, \theta(s)) \tag{2.18}$$

where i = 1, 2, or 3 depending on the initial angle of the ray leaving the source. Rays with i =1 leave the source a sufficiently large angle upward so they do not have a turning point and are never refracted downward. The case i =2 corresponds to the limiting ray that has a turning point at infinite height. Rays described with i = 3 have turning points at finite height and are alternately refracted downward and reflected upward at the ground. These are the rays trapped by the inversion.

Rays that initially are angled downward are given by (using the negative sign in (2.3))

$$r = F_i(s, \theta(s)) - F_i(z, \theta(s)) \tag{2.19}$$

for all three cases before they are reflected upward at the ground. The reflected waves are given by (using the positive sign)

$$r = F_i(z, \theta(s)) + F_i(s, \theta(s)) - 2\ F_i(0, \theta(s)) \tag{2.20}$$

in all three cases. Note that after reflection the i = 3 rays are refracted downward and reflected upward from the ground repeatedly. In the case of these i = 3 type rays four more forms exist. For rays that initially were angled upward (using the negative sign)

$$r = -F_3(z, \theta(s)) - F_3(s, \theta(s)) - 2\ n\ F_3(0, \theta(s)). \tag{2.21}$$

after they have been refracted downward and have been reflected n times from the ground. For rays that initially were angled upward and have been reflected upward n times from the ground and have not been refracted downward following that reflection

$$r = F_3(z, \theta(s)) - F_3(s, \theta(s)) -2\ n\ F_3(0, \theta(s)) \tag{2.22}$$

Thus an i = 3 type ray leaving the source upward is first described by (2.18) or (2.22) with n = 0 before it is refracted downward through a turning point. After it is refracted downward the first time it is described by (2.21) with n = 0. Following its first reflection from the ground it is given by (2.22) with n = 1, then by (2.21) with n =1 between refraction through a turning point and reflection, then (2.22) with n =2, etc.

For rays that initially were angled downward (using the positive sign)

$$r = F_3(z, \theta(s)) + F_3(s, \theta(s)) -2\ n\ F_3(0, \theta(s)) \tag{2.23}$$

after they have been reflected upward n times from the ground. For rays that were initially angled downward (using the negative sign)

$$r = - F_3(z, \theta(s)) + F_3(s, \theta(s)) -2\, n\, F_3(0, \theta(s)) \tag{2.24}$$

after they have been reflected n times from the ground and have been refracted downward through a turning point.

Thus an i = 3 ray that is initially angled downward at the source will first be described by (2.19) or (2.24) with n = 0 until it reflects from the ground, then by (2.20) or (2.23) with n = 1 between reflection and refraction through a turning point. Following the turning point and before the second reflection (2.24) with n = 1. Then by (2.23) with n = 2, etc.

The functions F_i are given by

$$F_1(z, \theta(s)) = \frac{\sqrt{(A\,z + B)(C\,z + D)}}{A} + \frac{E}{2\,A\sqrt{A\,C}} \ln\left(\frac{\phi + 1}{\phi - 1}\right) \tag{2.25}$$

for i = 1,

$$F_2(z, \theta(s)) = \frac{2}{3\,C\sqrt{B}}\,(C\,z + D)^{\frac{3}{2}} \tag{2.26}$$

for i = 2, and

$$F_3(z,\, \theta(s)) = \frac{\sqrt{(A\,z + B)(C\,z + D)}}{A} + \frac{E}{A\sqrt{-\,A\,C}}\,\mathrm{Tan}^{-1}\left(\frac{1}{\phi}\right)$$

(2.27)

for i = 3. As discussed above, the i = 1 case occurs for A greater then zero or for Cos θ(s) < Cos Θ (θ > Θ) where

$$\mathrm{Cos}\,\Theta = \sqrt{\frac{1 + \alpha\,s + \dfrac{\Delta T}{T_\infty}}{1 + \alpha\,s}}$$

(2.28)

Rays with values of the initial angle, θ(s), greater the value of the angle given by (2.28) then escape from the trapping effect of the inversion. The ray with A equal to zero or θ(s) = Θ is the limiting ray that has its turning point at infinity (the i = 2 case), while the rays with i =3 correspond to negative values of A or Cos θ(s) > CosΘ (θ(s) < Θ), and are the rays trapped by the ground. Rays with initial downward slopes can be divided in a similar manner but in all cases at least one reflection occurs before the ray escapes the trapping effect of the inversion, is the limiting case, or becomes trapped by the inversion.

Figure 2.3 is an example of the rays calculated from the above equations for the case of an inversion.

3.0 LAPSE CASE SOLUTION

The solution of the problem posed by equations (1.2) to (1.4) in case of $\Delta T > 0$, a lapse condition, was undertaken first. The general approach used in both the lapse and inversion cases was to first separated out a sinusoidal time dependence from the pressure, and then to Hankel transform the governing equations with respect to the horizontal distance from the source, r, to reduce the number of independent variables to one, the vertical height, z. This reduces the governing equation for the transformed independent variable to an ordinary differential equation for which an approximate solution can be obtained. This solution contains the Hankel transform variable, β, which replaced the horizontal distance in the transformed governing equation. The transformed solution must then be inverse transformed to return to physical space. Because of the complexity of the solution this inverse transform can not be carried out exactly and either an approximate inversion must be used or the inversion must be carried out numerically. Both approaches were used in the lapse case. In both of these approaches it is necessary to to interpret the Hankel transform variable as a complex variable and to continue the solutions off the real axis for the transform variable. This is not an intuitive process as the physical interpretation of the transform variable is lost off the real axis. This process will be discussed in detail below.

With the solution obtained for complex values of the transform variable attention will be turned to the inversion of the transformed solution. The methods used are the classical saddle point approach and a Fast Fourier Transform (FFT) based numerical method. These methods are describe in detail elsewhere and will be only described briefly here. Finally the results of these approaches will be described.

3.1 Transformation and approximate solution

The time dependence in the governing equation and boundary conditions, equation (1.3) and (1.4) can be removed by assuming

$$p(z, r, t) = e^{i\omega t}\, \overline{G}(z, r) \tag{3.1}$$

The Hankel transform or two-dimension Fourier transform for an axisymmetric function can be defined [6] as

$$G(z,\beta) = \int_0^\infty \overline{G}(z, r)\, r\, J_0(\beta r)\, dr \tag{3.2}$$

and the inverse transform as

$$\overline{G}(z, r) = \int_0^\infty G(z, \beta)\, \beta\, J_0(\beta r)\, d\beta \tag{3.3}$$

The use of transform methods in solving partial differential equations arises from the fact that an appropriate transform will convert a particular type of derivatives into an algebraic term expressed in terms of the transform variable (β in (3.2) and (3.3)) in place of the original physical independent variable (r in (3.2) and (3.3)). Thus the number of independent variables in the partial differential equation will be reduced by one and the transform variable acts only as a parameter in the transformed solution. In the case of Hankel transforms the radial dependence in the Laplacian operator expressed in cylindrical coordinates for a axisymmetric function is converted to an algebraic term, see [6] for more details. Applying (3.2) to (1.3) leads to

$$\frac{d^2G}{dz^2} + \left[\frac{\omega^2}{a_\infty^2} \frac{1 + \alpha z}{1 + \alpha z + \frac{\Delta T}{T_\infty}} - \beta^2 \right] G = - \frac{q}{2\pi} \delta(z\text{-}s) \tag{3.4}$$

The term on the right hand side represents the source. The homogeneous form of this equation would have a solution with an oscillating behavior if the term in square brackets was positive and an exponential behavior if it was negative. Thus the transition occurs when the term is zero or

$$\beta = \frac{\omega}{a_\infty} \sqrt{\frac{1 + \alpha z}{1 + \alpha z + \frac{\Delta T}{T_\infty}}} \tag{3.5}$$

Notice that if

$$\beta = \frac{\omega}{a_\infty} \sqrt{\frac{1 + \alpha s}{1 + \alpha s + \frac{\Delta T}{T_\infty}}} \text{ Cos } \theta(s) \tag{3.6}$$

then equations (3.5) and (2.14) are very similar and we can interpret (3.5) as giving the value of β that causes the turning point (the location of the transition from oscillatory to exponential behavior as well as a horizontal ray) to be located at the height z. But (3.6) remains to be be interpreted. From (2.1) and (2.2), however, one finds that β is equal to (ω/a_∞) times the cosine of the angle that a ray, which was initially at an angle $\theta(s)$, makes in the limit as height tends to infinity. Thus just as we have used $\theta(s)$ to identify a ray we can also use the Hankel transform variable β. This discussion emphasizes the close relationship between the model being developed and the ray description. These parallels will be also be pointed out below.

If height is nondimensionalized in (3.4) using the scale of the temperature gradient, α, a uniformly valid approximate solution to the resulting equation can be obtained, using the method presented by Nayfah [7], for large values of $\omega/(a_\infty\alpha)$ as

$$G = \frac{A}{\sqrt{g_z(z, \beta)}} h_1(\eta(z, \beta)) + \frac{B}{\sqrt{g_z(z, \beta)}} h_2(\eta(z, \beta)) \qquad (3.7)$$

where

$$g^{\frac{3}{2}}(z, \beta) = (1 + \alpha z + \frac{\Delta T}{T_\infty}) \sqrt{\frac{1 + \alpha z}{1 + \alpha z + \frac{\Delta T}{T_\infty}} - (\frac{a_\infty}{\omega} \beta)^2}$$

$$-\frac{1}{2}\frac{\Delta T}{T_\infty}\frac{1}{\sqrt{1-\left(\frac{a_\infty}{\omega}\beta\right)^2}}\ln\left(\frac{1-\phi}{1+\phi}\right)$$

(3.8)

$$\Phi=\sqrt{\frac{\dfrac{1+\alpha z}{1+\alpha z+\dfrac{\Delta T}{T_\infty}}-\left(\dfrac{a_\infty}{\omega}\beta\right)^2}{1-\left(\dfrac{a_\infty}{\omega}\beta\right)^2}}$$

(3.9)

$$\eta(z,\beta)=\left(\frac{3}{2}\lambda\right)^{\frac{3}{2}}g(z,\beta)$$

(3.10)

and

$$g_z(z,\beta)=\frac{\partial g(z,\beta)}{\partial z}$$

(3.11)

The modified Hankel functions h_1 and h_2 are defined in [8] by

$$h_1(\xi) = \left(\frac{2}{3}\right)^{\frac{1}{3}} \xi^{\frac{1}{2}} H^{(1)}_{\frac{1}{3}}\left(\frac{2}{3}\xi^{\frac{3}{2}}\right)$$

(3.12)

and

$$h_2(\xi) = \left(\frac{2}{3}\right)^{\frac{1}{3}} \xi^{\frac{1}{2}} H^{(2)}_{\frac{1}{3}}\left(\frac{2}{3}\xi^{\frac{3}{2}}\right)$$

(3.13)

A and B are constants that have to be determined. This solution is rather complex and is expressed in terms of unfamiliar functions. Several important features of this solution need to be discussed to understand it.

The functions $h_1(\xi)$ and $h_2(\xi)$ have a complicated behavior [8]. For real values of the argument both h_1 and h_2 yield a complex results which is oscillatory with an algebraic decay of the amplitude for increasing magnitude of the argument. The function h_1 can be shown to represent downward propagating waves (for $e^{i\omega t}$ as used here) and h_2 upward propagating waves. The oscillatory behavior also occurs for h_1 when the phase of the argument is equal to $2\pi/3$ and for h_2 when the phase of the argument is $-2\pi/3$. When the phase of the argument is $\pi/3$, h_1 decays exponentially and h_2 grows exponentially. When the phase is $-\pi/3$, h_2 decays exponentially and h_1 grows exponentially.

The solution of the point source problem is closely related to the plane wave problem discussed in [9]. In wave problems where transforms have been used the inverse transform can generally be interpreted as a superposition of plane waves over a range of angles. In the case of the Hankel transforms used here the limits of integration on β are from zero to infinity and β can be interpreted as ω/a_∞ Cos θ_∞ where θ_∞ is the angle between a ray and the horizontal in the limit of height tending to infinity as given in (3.6). Thus a value of β of zero corresponds to a ray which is propagating vertically upward at infinite height, a value of β of ω/a_∞ corresponds to a horizontal ray at infinite height, and a value of β of $\omega/a_\infty \{[\ 1 + \alpha z\ ] / [\ 1 + \alpha z + \Delta T/T_\infty\ ]\}^{1/2}$ corresponds to a ray with imaginary slope at infinity such that its turning point (point of horizontal slope) is located at the height z. Based on this description the waves group themselves into several different forms.

The first group, $0 \leq \beta \leq \beta_0 = \omega/a_\infty \{\ 1\ / [\ 1 + \Delta T/T_\infty\ ]\}^{1/2}$, are waves with their turning points at or below the ground surface and thus are actually reflected at the surface. These waves plus their reflections constitute the first group. The wave with $\beta = \beta_0$ grazes the surface and is the limiting ray between the reflected and refracted rays. The second group, $\beta_0 \leq \beta \leq \beta_z = \omega/a_\infty \{[\ 1 + \alpha z\ ] / [\ 1 + \alpha z + \Delta T/T_\infty\ ]\}^{1/2}$, are waves with a turning point above the ground and below the observer height z. The waves in this group consist of those leaving the source in the range of angles described by (3.6) and their continuation after they have been refracted upward. The third group has $\beta_z \leq \beta \leq \beta_s = \omega/a_\infty \{[\ 1 + \alpha s\ ] / [\ 1 + \alpha s + \Delta T/T_\infty\ ]\}^{1/2}$. These waves have their turning point above the observer and below the source. At the observers location these waves should yield a exponentially decaying solution. The waves in this group consist of those leaving the source in the appropriate range of angles and their continuation following refraction upward. These three groups of rays can all be seen in a ray diagram for a point source and all initially are propagating downward from the source. In addition there are waves propagating upward initially. These are in the range $0 \leq \beta \leq \beta_s$ but differ from the first groups in that they do not originate as downward propagating waves that are reflected or refracted upward. Thus the "reflection" coefficient is missing from these waves.

In addition to the above groups that can be seen in a wave diagram for a point source, there are several types that are necessary for the superposition given by (3.3) where β ranges from zero to infinity, but do not occur in a ray diagram for a point source. Group five consists has $\beta_s \leq \beta \leq \omega/a_\infty$, these waves have the turning point above the source. In addition there are waves with $\omega/a_\infty < \beta$, these have no physical interpretation and correspond to complex angles at infinity.

With these concepts let us proceed to the mathematical solution to the problem. The function $g^{3/2}$ given in (3.8) contains four branch points, two at $\beta = \pm \beta_z$ and two at $\beta = \pm \omega/a_\infty$. The negative branch points are not significant and will not be discussed. On the real β axis, for $0 \leq \beta < \beta_z$, $g^{3/2}(z, \beta)$ is real. For $\beta_z < \beta < \omega/a_\infty$ $g^{3/2}(z,\beta)$ is positive and imaginary, and for $\beta > \omega/a_\infty$ $g^{3/2}(z,\beta)$ has a phase of $-\pi$ at $\beta = \omega/a_\infty$ and tends to a phase of $-\pi/2$ as β tends toward infinity. This is shown in Figure 3.1. The branch line for $g(z,\beta) = (g^{3/2}(z,\beta))^{2/3}$ can be chosen to be on the line where the phase of $g^{3/2}(z,\beta)$ is $-\pi$. This line extends from the first branch point at β_z to the second at ω/a_∞ and encloses a small region above the positive real axis, see Figure 3.2. The branches chosen for $g(z,\beta)$ yield a phase of zero for $0 \leq \beta < \beta_z$, $\pi/3$ for $\beta_z < \beta < \omega/a_\infty$, and varying from $-2\pi/3$ to $-\pi/3$ for $\beta > \omega/a_\infty$.

Now by considering three cases the various forms of the solution can be obtained. These are shown in Figure 3.3. The first case is a wave with the turning point below the surface, $0 \leq \beta \leq \beta_0$. The second has the turning point below the source but above the surface, $\beta_0 \leq \beta \leq \beta_s$. In this case if the receiver is below the turning point then $\beta > \beta_z$, if it is above the turning point then $\beta < \beta_z$. The third case has $\beta_s \leq \beta \leq \omega/a_\infty$ and the turning point is above the source. Again if the turning point is above the receiver then $\beta > \beta_z$, if the receiver is above the turning point then $\beta < \beta_z$. Note that these waves do not appear in a point source ray diagram but are needed to complete the solution.

The solutions corresponding to the cases given above require the determination of the constants in (3.7). To do this a set of conditions are required. As result of the source terms in (3.4) the solutions separate into at least two forms, one for the region below the source and one above. The radiation condition, requiring outgoing waves in the limit as height tends to infinity, requires A to be zero above the turning point for $z > s$. At the source height, $z = s$, the solution must be continuous

$$\lim_{z \to s_+} G(z,\beta) = \lim_{z \to s_-} G(z,\beta) \tag{3.14}$$

and must satisfy

$$\lim_{z\to s_-}\left[\frac{\partial G}{\partial z}\right] - \lim_{z\to s_+}\left[\frac{\partial G}{\partial z}\right] = \frac{q}{2\pi}$$

(3.15)

which is obtained by integrating (3.4) from $z = s - \varepsilon$ to $z = s + \varepsilon$ and taking the limit $\varepsilon\to 0$. At a turning point continuity is required,

$$\lim_{z\to z_{tp-}} G(z,\beta) = \lim_{z\to z_{tp+}} G(z,\beta)$$

(3.16)

At the ground surface, $z = 0$, the required condition is the normal impedance condition (1.4) which can now be expressed as

$$G = \frac{i\,Z}{\omega\rho_o}\,\frac{\partial G}{\partial z}$$

(3.17)

Using these conditions, equation (3.4), and the physical situations presented in Figure 3.2 the following solutions can be obtained. For $z > s$

$$G = K\, h_2(\eta(z,\beta))\,[\, h_1(\eta(s,\beta) + R_o\, h_2(\eta(s,\beta))\,]$$

(3.18)

for $\omega/a_\infty > \beta_z > \beta_s > \beta_o > \beta > 0$,

$$G = K\, h_2(\eta(z,\beta))\, [\, h_1(\eta(s,\beta) + R_1\, h_2(\eta(s,\beta))\,] \tag{3.19}$$

for $\omega/a_\infty > \beta_z > \beta_s > \beta > \beta_o > 0$,

$$G = K\, h_2(\eta(z,\beta))\, e^{i\pi/3}\, R_1\, [\, h_1(\eta(s,\beta) + R_0\, h_2(\eta(s,\beta))\,] \tag{3.20}$$

for $\omega/a_\infty > \beta_z > \beta > \beta_s > \beta_o > 0$ and

$$\begin{aligned} G = K\, [\, h_1(\eta(z,\beta))\, e^{i\pi/3} + R_0\, h_2(\eta(z,\beta))\,]\, e^{i\pi/3} \\ R_1\, [\, h_1(\eta(s,\beta) + R_0\, h_2(\eta(s,\beta))\,] \end{aligned} \tag{3.21}$$

for $\omega/a_\infty > \beta > \beta_z > \beta_s > \beta_o > 0$. For $z < s$

$$G = K\, h_2(\eta(s,\beta))\, [\, h_1(\eta(z,\beta) + R_o\, h_2(\eta(z,\beta))\,] \tag{3.22}$$

for $\omega/a_\infty > \beta_s > \beta_z > \beta_o > \beta > 0$,

$$G = K\, h_2(\eta(s,\beta))\, [\, h_1(\eta(z,\beta) + R_1\, h_2(\eta(z,\beta))\,] \tag{3.23}$$

for $\omega/a_\infty > \beta_s > \beta_z > \beta > \beta_o > 0$,

$$G = K\, h_2(\eta(s,\beta))\, e^{i\pi/3}\, R_1\, [\, h_1(\eta(z,\beta) + R_0\, h_2(\eta(z,\beta))\,] \tag{3.24}$$

for $\omega/a_\infty > \beta_s > \beta > \beta_z > \beta_o > 0$ and

$$G = K\, [\, h_1(\eta(s,\beta))\, e^{i\pi/3} + R_0\, h_2(\eta(s,\beta))\,]\, e^{i\pi/3}$$
$$R_1\, [\, h_1(\eta(z,\beta) + R_0\, h_2(\eta(z,\beta))\,] \tag{3.25}$$

for $\omega/a_\infty > \beta > \beta_s > \beta_z > \beta_o > 0$. Where

$$K = K(g_z(z,\beta),\, g_z(s,\beta)) = \frac{q}{12\, i\, \lambda^{\frac{2}{3}}}\, \frac{1}{\sqrt{g_z(z,\beta)}}\, \frac{1}{\sqrt{g_z(s,\beta)}} \tag{3.25}$$

$$R_o = -\,\frac{\tau\, h_1(\eta(0,\beta)) + i\, \psi\, h_1'(\eta(0,\beta))}{\tau\, h_2(\eta(0,\beta)) + i\, \psi\, h_2'(\eta(0,\beta))} \tag{3.26}$$

$$\tau = \alpha\, \lambda - \frac{i}{2}\, \frac{Z}{\rho_o a_\infty}\, \frac{g_{zz}(0,\beta)}{g_z(0,\beta)} \tag{3.27}$$

$$\psi = \frac{Z}{\rho_o a_\infty} \left(\frac{3\lambda}{2}\right)^{\frac{2}{3}} g_z(0,\beta)$$

(3.28)

and

$$R_1 = \frac{e^{-i\frac{\pi}{6}}}{e^{i\frac{\pi}{6}} - i R_o}$$

(3.29)

The solutions obtained above are for real values of β, but as discussed above β must be interpreted as a complex variable. The boundaries between solutions off the real axis must be chosen as the branch lines used for calculating the function g(z,β), g(s,β) and g(0,β) from $g^{3/2}(z,\beta)$, etc. as were discussed above. On crossing these branch lines it should be noted that the phase of g(z,β), g(s,β) and g(0,β) discontinuously jumps from -2π/3 to 2π/3 and the phase of $g_z(z,\beta)$, $g_z(s,\beta)$ and $g_z(0,\beta)$ increases by -2π/3 (since it contains the root of g in the denominator).

Reference [8] presents some results for the modified Hankel functions that are useful for this type of behavior:

$$h_2(\eta\, e^{i\frac{2}{3}\pi}) = e^{i\frac{\pi}{3}}(\, e^{i\frac{\pi}{3}} h_1(\eta) + h_2(\eta)\,)$$

(3.30)

and

$$h_1(\,\eta\, e^{i\frac{2}{3}\pi}\,) = -\, h_2(\eta)$$

(3.31)

Using these relations the solution as given by equation (3.18) to (3.25) can be rewritten and the regions of validity determined. For z > s these are

$$G = K(g_z(z,\beta),\, g_z(s,\beta))\,\{h_2(\eta(z,\beta))\,[\, h_1(\eta(s,\beta)) + R_o(g(0,\beta))\, h_2(\eta(s,\beta))\,]\}$$

(3.32)

in region A of β-space as given in Figure 3.4

$$G = K(g_z(z,\beta),\, g_z(s,\beta))\,\{h_2(\eta(z,\beta))\,[\, h_1(\eta(s,\beta)) + R_o(g(0,\beta)\, e^{i\,2\pi/3})\, h_2(\eta(s,\beta))\,]\}$$

(3.33)

in region B

$$G = K(g_z(z,\beta)e^{-i\,2\pi/3},\, g_z(s,\beta))\,\{h_2(\eta(z,\beta))\,[\, h_1(\eta(s,\beta)\, e^{i\,2\pi/3}) + R_o(g(0,\beta)\, e^{i\,2\pi/3})\, h_2(\eta(s,\beta)\, e^{i\,2\pi/3})\,]\}$$

(3.34)

in region C

$$G = K(g_z(z,\beta)e^{-i\,2\pi/3}, g_z(s,\beta)\, e^{-i\,2\pi/3})\, \{h_2(\eta(z,\beta)\, e^{i\,2\pi/3})\, [\, h_1(\eta(s,\beta)\, e^{i\,2\pi/3}) + R_o(g(0,\beta)\, e^{i\,2\pi/3})\, h_2(\eta(s,\beta)\, e^{i\,2\pi/3})\,]\} \tag{3.35}$$

in region D. For $z < s$

$$G = K(g_z(z,\beta), g_z(s,\beta))\, h_2(\eta(s,\beta))\, [\, h_1(\eta(z,\beta)) + R_o\, (g(0,\beta))\, h_2(\eta(z,\beta))\,] \tag{3.36}$$

in region E

$$G = K(g_z(z,\beta), g_z(s,\beta))\, h_2(\eta(s,\beta))\, [\, h_1(\eta(z,\beta)) + R_o\, (g(0,\beta)\, e^{i\,2\pi/3})\, h_2(\eta(z,\beta))\,] \tag{3.37}$$

in region F

$$G = K(g_z(z,\beta), g_z(s,\beta)\, e^{-i\,2\pi/3})\, h_2(\eta(s,\beta))\, [\, h_1(\eta(z,\beta)\, e^{i\,2\pi/3}) + R_o\, (g(0,\beta)\, e^{i\,2\pi/3})\, h_2(\eta(z,\beta)\, e^{i\,2\pi/3})\,] \tag{3.38}$$

in region G and

$$G = K(g_z(z,\beta)e^{-i\,2\pi/3}, g_z(s,\beta)\, e^{-i\,2\pi/3})\, h_2(\eta(s,\beta)\, e^{i\,2\pi/3})\, [\, h_1(\eta(z,\beta)\, e^{i\,2\pi/3}) + R_o\, (g(0,\beta)\, e^{i\,2\pi/3})\, h_2(\eta(z,\beta)\, e^{i\,2\pi/3})\,] \tag{3.39}$$

in region H. To obtain these results it is necessary to recognize that

$$R_1(g(0,\beta)) = R_o(g(0,\beta)\, e^{i\frac{2}{3}\pi})$$

(3.40)

From these results and the description of the behavior of the function g(z,β) (and therefore η) at the branch lines it should be clear that the solution is continuous at the branch lines even with g(z,β) being discontinuous. This transformed solution must now be inverted.

3.2 Lapse Results

Two methods were used to approximately invert the Hankel transform contour integration which lead to the saddle point method in the insonified region of physical space and a FFT based numerical method. Neither of these methods will be described in detail as the basic method is well known in both cases and the specific application has been described in detail elsewhere.

Both of these approaches are based on the concept that although the inversion integral (3.3) is defined as being carried out along the real axis, the residue theorem of complex variables [10] allows the path of integration to be changed provided that there are no poles of the integrand between the original and modified paths. If poles exist then additional terms must be included with the integral along the modified path. In the case of an isothermal atmosphere the additional term due to the pole leads to the surface wave term. In the lapse case the only possible pole is the due to the denominator of the term multiplying the upward going wave (the reflection coefficient in the case where a reflection occurs) being equal to zero. These case has not been completely examined but in the limit of ΔT equal to zero it reproduces the surface wave term. Thus one clearly expects to see a similar behavior in the case of weak lapse condition. This surface wave like-behavior has not been investigated beyond the point described above and has not been included in the results given below.

3.2.1 Contour Integration-Saddle Point Method

Integrals of the form

$$I(\xi) = \int_C e^{i \xi f(\beta)} d\beta$$

(3.41)

can be approximately evaluated by the saddle point method if the path of integration C is such that the ends of the path do not significantly contribute to the integral, ξ is a large parameter, and $f(\beta)$ has a point where the first derivative is zero. Complex variable theory indicates that a function can not have maximum in the region where it is analytic and a point where its first derivative is zero must be a saddle point [10]. At a saddle point if the path of integration follows the line of constant real part of $f(\beta)$ then the imaginary part $f(\beta)$ either increases or decreases at a maximum rate. If the path of integration follows the line of constant imaginary part of $f(\beta)$ that passes through the saddle point then the real part of $f(\beta)$ increases or decreases at a maximum rate. If we choose to follow a line of constant real part of $f(\beta)$ through the saddle point in the direction such that the imaginary part increases at the maximum rate then the magnitude of the integrand decrease rapidly as we move away from the saddle point. If ξ is large then the only significant part of the integral is near the saddle point and the integral can be approximated by using the first two non-zero terms in the Taylor series for $f(\beta)$ yielding the well known results given in [11].

This method works well when a saddle point exists. However when one does not exist then an approximate integration can be carried out as described in detail by Ma [12].

To apply this method to the integrals given by (3.3) with (3.32) to (3.39) both the

Hankel functions and the modified Hankel functions must be replaced by their asymptotic expansions and the integral regrouped to extend from negative infinity to positive infinity. When this is done the result contains twenty terms since the integrand is different between in the various regions in β-space and in each part of the G function contains two terms. Thus writing out only the terms of interest for z > s yields

$$\begin{aligned}
\overline{G}(z,\beta) = \ldots &+ \int_{0}^{\beta_o} K_6(z,\beta)\, e^{-i\{\beta r + \lambda[g^{3/2}(z,\beta) - g^{3/2}(s,\beta)] - \pi/2\}}\, d\beta \\
&+ \int_{\beta_o}^{\beta_s} K_7(z,\beta)\, e^{-i\{\beta r + \lambda[g^{3/2}(z,\beta) - g^{3/2}(s,\beta)] - \pi/2\}}\, d\beta \\
&+ \ldots \\
&+ \int_{0}^{\beta_o} K_{16}(z,\beta)\, e^{-i\{\beta r + \lambda[g^{3/2}(z,\beta) + g^{3/2}(s,\beta) - 2g^{3/2}(0,\beta)] - \pi/2\}}\, d\beta \\
&+ \int_{\beta_o}^{\beta_s} K_{17}(z,\beta)\, e^{-i\{\beta r + \lambda[g^{3/2}(z,\beta) + g^{3/2}(s,\beta)] - \pi/6\}}\, d\beta \\
&+ \ldots
\end{aligned} \tag{3.42}$$

and for z< s

$$\overline{G}(z,r) = \dots + \int_{0}^{\beta_o} C_6(z,r)\, e^{-i\{\beta r + \lambda[g^{3/2}(s,\beta) - g^{3/2}(z,\beta)] - \pi/2\}} d\beta$$

$$+ \int_{\beta_o}^{\beta_z} C_7(z,\beta)\, e^{-i\{\beta r + \lambda[g^{3/2}(s,\beta) - g^{3/2}(z,\beta)] - \pi/2\}} d\beta$$

$$+ \dots$$

$$+ \int_{0}^{\beta_o} C_{16}(z,\beta)\, e^{-i\{\beta r + \lambda[g^{3/2}(z,\beta) + g^{3/2}(s,\beta) - 2\,g^{3/2}(0,\beta)] - \pi/2\}} d\beta$$

$$+ \int_{\beta_o}^{\beta_z} C_{17}(z,\beta)\, e^{-i\{\beta r + \lambda[g^{3/2}(z,\beta) + g^{3/2}(s,\beta)] - \pi/6\}} d\beta$$

$$+ \dots \tag{3.43}$$

To find the saddle points the arguement of of exponential term must be differentiated with respect to β and set equal to zero. On differentiating one finds

$$\frac{\partial}{\partial\beta}\left(\lambda\, g^{\frac{3}{2}}(z,\beta)\right) = -\, F\left(z,\, \mathrm{Cos}^{-1}\left\{\frac{a_\infty}{\omega}\sqrt{\frac{1+\alpha s+\frac{\Delta T}{T_\infty}}{1+\alpha s}}\,\beta\right\}\right) \tag{3.44}$$

where F(z,θ(s)) is the function defined in (2.10) and used to describe the rays. Differentiating the arguments of the exponentials then yields equations (2.6) to (2.9), the equations defining the rays. Thus the saddle points associated with a particular point in physical space (in the insonified region) correspond to the values of β (or θ(s) where the two are related by (3.6)) that defines the two rays passing through the point in physical space. Just as the rays were interpreted as upward-going-direct waves, refracted waves, etc. the terms in (3.42) and (3.43) also have the same interpretations. Determination of the saddle point values then first requires determination of the types of waves present at a particular physical location and then solution of the appropriate two of equations (2.6) to (2.9). Once the location of the saddle point has been determined the classical results may be applied. A computer program for carrying out this procedure and the resulting equations to approximate the inversion integral have been given in detail by Cheng [13] and will not be repeated here. A typical result is shown in Figure 3.5.

As the physical location of the receiver moves into the acoustic shadow real values of β or θ(s) cease to exist. Ma [12] has suggested an approximate approach which is also based on contour integration. In this approach the inversion integral between β_z or β_s and ω/a_∞ is carried out numerically and the remainder of the integral extending from negative infinity and to positive infinity are carried out in a manner similar to the saddle point method. The integral carried out numerically physically represents the contributions of the exponentially decaying disturbances due to waves with turning points above the receiver. Ma [12] also incorporated into his program Cheng's saddle point method for the insonified region with some changes. A typical result is shown in Figure 3.6.

Both of these methods suffer from discontinuities as the receiver passes from a region where the receiver senses a direct and a reflected wave to one where a direct and a refracted wave would occur. This results not from the transformed solution which is continuous, but from the large argument approximation that must be made to obtain the saddle point form (3.41) and from the fact that the argument becomes zero at most of the boundaries. As a result of these intrinsic problems with the saddle point method a numerical approach was then applied.

The saddle point method has the appeal of a physical interpretation of the mathematical steps and results. A purely numerical method loses that interpretation and the physical insight that comes from it.

3.2.2 Numerical Integration Method

The numerical approach used was developed by Richards and Attenborough [14] and was applied to the present case by Lloyd [15]. The method approximates the inversion integral by using a Fast Fourier Transform (FFT) algorithm. To obtain an integral suitable for the use of the FFT algorithm the Bessel function containing the horizontal distance dependence must be approximated by its asymptotic expansion. Three other modifications are then carried out. First, the integration path is modified to be above the real axis (Richards and Attenborough's original approach was to integrate below the axis but they also assumed $e^{-i\omega t}$.), this avoids the discontinuities at the branch points but requires integration up the imaginary axis. Second, the integrand is modified to make the integral along the imaginary axis zero, but, as claimed by Richards and Attenborough, to not change the result. Finally an approximate term is added to account for the finite upper limit and to approximate the integral to infinity.

This approach is described in detail by Lloyd [15] and Figure 3.7 is a typical result. Disadvantages of this approach are the large amount of computer time required and that a entire horizontal profile must be obtained at each receiver height. Thus to obtain a vertical profile many time consuming computer runs must be made and one point out of several thousand points is actually used from each run. This approach clearly does not contain the discontinuities present in the saddle point method. Figure 3.8 compares the saddle point method and the purely numerical method. The agreement is excellent in the insonified region with the exception of the region very near the shadow boundary. The agreement is good in the initial sound level decrease as the shadow boundary is crossed but the saturation region deep in the shadow is not the same for the two methods.

The numerical method often results in oscillations in the sound level at large distances from the source, this appears to be an artifact of the numerical inversion method and is dependent on the parameters of the inversion scheme. Also as very large distances are approached the calculated sound level often increases this is clearly due to the numerical inversion method. These points are further discussed by Lloyd [15] and a listing of Lloyd's program is given in Appendix A.

4.0 INVERSION SOLUTION

The inversion solution, $\Delta T < 0$, follows very closely along the lines of the lapse solution. The governing equation and boundary conditions, (1.2) to (1.4), are identical in the two cases and the approach using Hankel transforms is also the same. Again the solution requires analytic continuation off the real β axis. This has proved to be difficult and lead to several errors initially. (The solution given in the Sixth Semiannual Report [18] are incorrect.)

Although the solution can be discussed in terms of rays and the closely related saddle point method, this approach has not been used to approximately evaluate the solution in the inversion case. In the lapse case only two rays, at most, pass through a given point. In the inversion case, at large distances from the source, many rays may pass through a given point due to the "trapping" effect of the inversion. Since the saddle point method requires all of these rays and their corresponding saddle points to be located, and this is the most difficult part of the method, the approach becomes impractical. Thus only the purely numerical method of inversion has been used.

4.1 Transformation and approximate solution

The time dependence is removed from (1.3) as in (3.1) and the resulting equation Hankel transformed using (3.2) to obtain (3.4). Again the location (in terms of the transform variable β) of the transition from oscillating to exponential behavior is given by (3.5). However, since $\Delta T/T_\infty < 0$ the transition is at a value of β greater then ω/a_∞. Using (3.6) and comparing results to those of Section 2.2 one can interpret the solution in the range $0 \leq \beta < \omega/a_\infty$ as representing the rays that will escape the trapping effect of the inversion. These rays either go directly from the source to infinite heights or go from the source to the ground where a reflection occurs and then go to infinite heights. In the range $\omega/a_\infty \leq \beta \leq \beta_s$ the solution represents rays that are trapped by the inversion. The transition given by (3.5) applies to rays in this range. Beyond this range the rays do not occur in a ray diagram.

The solution of (3.4) can again be approximated by (3.7) and (3.9) through (3.11). Equation (3.8) must modified by a negative sign on the right hand side yielding

$$g^{\frac{3}{2}}(z,\beta) = -\left(1 + \alpha z + \frac{\Delta T}{T_\infty}\right)\sqrt{\frac{1+\alpha z}{1+\alpha z+\frac{\Delta T}{T_\infty}} - \left(\frac{a_\infty}{\omega}\beta\right)^2}$$

$$+\frac{1}{2}\frac{\Delta T}{T_\infty}\frac{1}{\sqrt{1-\left(\frac{a_\infty}{\omega}\beta\right)^2}}\ln\left(\frac{1+\Phi}{1-\Phi}\right) \tag{4.1}$$

This change is necessary since the region of oscillatory solution is below the turning point in the inversion case while it was above it in the lapse case (see Nayfeh [7]). In the inversion case $\Delta T/T_\infty < 0$ and thus $\Phi > 1$ for $0 \leq \beta < \omega/a_\infty$. It is convenient to note that we may rewrite the logarithm term in this case as

$$\frac{1}{2}\ln\left(\frac{1+\Phi}{1-\Phi}\right) = \mathrm{Tanh}^{-1}\left(\frac{1}{\Phi}\right) + i\frac{\pi}{2} \tag{4.2}$$

to clearly indicate the choice of the branch of logarithm, $\ln(-1) = +i\,\pi$, the opposite of that in the lapse case. Thus, in the range $0 \le \beta < \omega/a_\infty$ $g^{3/2}(z,\beta)$ ranges from slightly below the negative real axis to infinity along the negative imaginary axis. For values of β between ω/a_∞ and β_z ranges along the positive real axis from infinity to zero. For real β greater than β_z values of $g^{3/2}(z,\beta)$ are along the positive imaginary axis, see Figure 4.1. As in the lapse case the boundary between these regions are branch points of the function $g^{3/2}(z,\beta)$ with the branch lines extending downward from the branch points for $\mathrm{Re}(\beta) > 0$ in β-space.

The argument of the Hankel functions involves $g(z,\beta) = (g^{3/2}(z,\beta))^{2/3}$ and again the branches must be chosen with care. For $0 \le \beta < \omega/a_\infty$ $g(z,\beta)$ is chosen such that its phase ranges from slightly greater then zero (or 2π) at $\beta = 0$ to $\pi/3$ as the branch point at $\beta = \omega/a_\infty$ is approached from values of β less then ω/a_∞. In this region the two modified Hankel functions have an oscillatory and exponential growth or decay behavior with with one (h_1) representing upward traveling and decaying waves and the other (h_2) downward traveling, growing waves. In the range $\omega/a_\infty < \beta < \beta_z$ and $g^{3/2}(z,\beta)$ is real and positive. The function $g(z,\beta)$ is chosen to be on the line with phase

$-2\pi/3$. In this case the modified Hankel function h_1 represents both upward and downward traveling waves, a standing wave-like phenomena, and h_2 a downward traveling wave.

The branch line for $g(z,\beta) = (g^{3/2}(z,\beta))^{2/3}$ needs to be defined to extend these solutions off of the real axis. Lines of constant phase of $g^{3/2}(z,\beta)$ run from the first branch point to the second in a manner similar to the lapse case, but with the order of the branch points reversed. Figure 4.2 shows the behavior of these lines of constant phase. Again a line of constant phase is a convenient branch line.

If the line where the phase of $g^{3/2}(z,\beta)$ equals $-\pi/2$ is chosen as the branch line for $g(z,\beta) = (g^{3/2}(z,\beta))^{2/3}$ then the phase of $g(z,\beta)$ can be made to agree with the desired values on the real axis as described above. In addition for real β and $\beta_z < \beta$, $g(z,\beta)$ has a constant phase of $\pi/3$ with $h_1(\eta(z,\beta))$ having a decaying exponential behavior for increasing z and representing the contribution of waves with a turning point below the receiver's height to the total pressure field.

Using the conditions given in (3.14) to (3.17) and the physical descriptions of the type of waves that occur in each situation the following solutions can be obtained. For $z > s$

$$G = K[\, h_1(\eta(s,\beta)) + R_0\, h_2(\eta(s,\beta))\,]\, h_2(\eta(z,\beta)) \tag{4.3}$$

which is identical to (3.18) for $\beta_0 > \beta_s > \beta_z > \omega/a_\infty > \beta > 0$,

$$G = - K \, [\, h_1(\eta(s,\beta)) \, / \, R_o + h_2(\eta(s,\beta)) \,] \, h_1(\eta(z,\beta) \tag{4.4}$$

for $\beta_o > \beta_s > \beta_z > \beta > \omega / a_\infty > 0$,

$$G = - K \, e^{i\frac{2\pi}{3}} \, [\, h_1(\eta(s,\beta)) \, / \, R_o + h_2(\eta(s,\beta)) \,] \, h_2(\eta(z,\beta) \tag{4.5}$$

for $\beta_o > \beta_s > \beta > \beta_z > \omega / a_\infty > 0$ and

$$G = K \, [\, h_1(\eta(s,\beta)) + R_2 \, h_2(\eta(s,\beta)) \,] \, h_2(\eta(z,\beta)) \tag{4.6}$$

for $\beta_o > \beta > \beta_s > \beta_z > \omega / a_\infty > 0$. For $s > z$

$$G = K \, [\, h_1(\eta(z,\beta)) + R_o \, h_2(\eta(z,\beta)) \,] \, h_2(\eta(s,\beta)) \tag{4.7}$$

the same as (3.22) for $\beta_o > \beta_z > \beta_s > \omega / a_\infty > \beta > 0$,

$$G = - K \, [\, h_1(\eta(z,\beta)) \, / \, R_o + h_2(\eta(z,\beta)) \,] \, h_1(\eta(s,\beta) \tag{4.8}$$

for $\beta_o > \beta_z > \beta_s > \beta > \omega / a_\infty > 0$,

$$G = - K e^{i\frac{2\pi}{3}} [h_1(\eta(z,\beta)) / R_o + h_2(\eta(z,\beta))] h_2(\eta(s,\beta) \tag{4.9}$$

for $\beta_o > \beta_z > \beta > \beta_s > \omega/a_\infty > 0$ and

$$G = K [h_1(\eta(z,\beta)) + R_2 h_2(\eta(z,\beta))] h_2(\eta(s,\beta)) \tag{4.10}$$

for $\beta_o > \beta > \beta_z > \beta_s > \omega/a_\infty > 0$. Here K and R_o, and R_1 are defined by (3.25) and (3.26) and

$$R_2 = e^{-i\frac{\pi}{3}} \left(\frac{R_o - e^{-i\frac{\pi}{3}}}{R_o} \right) \tag{4.11}$$

As discussed above these solutions must be continued off the real axis. As in the lapse case the boundaries are chosen as the branch lines for calculating $g(z,\beta)$, $g(s,\beta)$ and $g(0,\beta)$ from $g^{3/2}(z,\beta)$, etc. On crossing these branch lines the phase of $g(z,\beta)$, $g(s,\beta)$ and $g(0,\beta)$ jumps discontinuously from $\pi/3$ to $-\pi$ and $g_z(z,\beta)$, $g_z(s,\beta)$ and $g_z(0,\beta)$ jumps discontinuously by $2\pi/3$. Using (3.30) and (3.31) solutions (4.3) through (4.10) can be continued off the axis as

$$G = K(g_z(z,\beta), g_z(s,\beta)) \,[\, h_1(\eta(s,\beta)) + R_o(g(0,\beta))\, h_2(\eta(s,\beta)) \,]\, h_2(\eta(z,\beta)) \tag{4.12}$$

in region A of Figure 4.3

$$G = K(g_z(z,\beta)\, e^{-i\frac{2\pi}{3}}, g_z(s,\beta)\, e^{-i\frac{2\pi}{3}}) \,[\, h_1(\eta(s,\beta)\, e^{i\frac{4\pi}{3}}) + R_o(g(0,\beta)\, e^{i\frac{4\pi}{3}})\, h_2(\eta(s,\beta)\, e^{i\frac{4\pi}{3}})]\, h_2(\eta(z,\beta)\, e^{i\frac{4\pi}{3}}) \tag{4.13}$$

in region B,

$$G = K(g_z(z,\beta), g_z(s,\beta)\, e^{-i\frac{2\pi}{3}}) \,[\, h_1(\eta(s,\beta)\, e^{i\frac{4\pi}{3}}) + R_o(g(0,\beta)\, e^{i\frac{4\pi}{3}})\, h_2(\eta(s,\beta)\, e^{i\frac{4\pi}{3}})]\, h_2(\eta(z,\beta)) \tag{4.14}$$

in region C and

$$G = K(g_z(z,\beta), g_z(s,\beta)) \,[\, h_1(\eta(s,\beta)) + R_o(g(0,\beta)\, e^{i\frac{4\pi}{3}})\, h_2(\eta(s,\beta)) \,]\, h_2(\eta(z,\beta)) \tag{4.15}$$

in region D. For $z < s$

$$G = K(g_z(z,\beta), g_z(s,\beta)) \,[\, h_1(\eta(z,\beta)) + R_o(g(0,\beta))\, h_2(\eta(z,\beta)) \,]\, h_2(\eta(s,\beta)) \tag{4.16}$$

in region E,

$$G = K(g_z(z,\beta)\, e^{-i\frac{2\pi}{3}}, g_z(s,\beta)\, e^{-i\frac{2\pi}{3}})\, [\, h_1(\eta(z,\beta)\, e^{i\frac{4\pi}{3}})$$
$$+ R_o(g(0,\beta)\, e^{i\frac{4\pi}{3}})\, h_2(\eta(z,\beta)\, e^{i\frac{4\pi}{3}})]\, h_2(\eta(s,\beta)\, e^{i\frac{4\pi}{3}}) \tag{4.17}$$

in region F,

$$G = K(g_z(z,\beta)\, e^{-i\frac{2\pi}{3}}, g_z(s,\beta))\, [\, h_1(\eta(z,\beta)\, e^{i\frac{4\pi}{3}})$$
$$+ R_o(g(0,\beta)\, e^{i\frac{4\pi}{3}})\, h_2(\eta(s,\beta))]\, h_2(\eta(z,\beta)\, e^{i\frac{4\pi}{3}}) \tag{4.18}$$

in region G and

$$G = K(g_z(z,\beta), g_z(s,\beta))\, [\, h_1(\eta(z,\beta)) + R_o(g(0,\beta)\, e^{i\frac{4\pi}{3}})\, h_2(\eta(z,\beta))\,]\, h_2(\eta(s,\beta)) \tag{4.19}$$

in region H.

From these results and the description of the behavior of the function g(z,β) (and therefore η) at the branch lines it should be clear that the solution is continuous at the branch lines even with g(z,β) being discontinuous. The transformed solution must now be inverted using the numerical method developed by Richards and Attenborough [14].

4.2 Inversion Case Results

As was discussed above the numerical method originally developed by Richards and Attenborough [14] was used to invert the Hankel transformed solution in the case of the inversion. This was due to the fact that many rays pass through a given point in physical space in the case of an inversion and the number of saddle points which exist equals the number of rays passing though that point. Since finding the saddle points is the most difficult and time consuming part of that method the approach appeared impractical in this case. The numerical method used is identical to that of the lapse case as described by Lloyd [15].

Only a limited number of cases have been run to date using the solution described in Section 4.1 and the numerical inversion technique. Figure 4.4 shows a typical case. The results generally show an interference pattern with 6 dB/doubling of distance decay out to distances of the order of thirty meters and a more complicated behavior beyond that distance but with no significant change in the rate of decay. This latter result is somewhat unexpected from qualitative arguments. Experimental data for propagation under inversion conditions is quite limited, with the data presented by Sutherland and Brown [16] being the major set. However, this set contains only seven measurements at a fixed height over a 675 meter distance. No direct comparisons have been made but the data also shows what appears to be a 6 dB/doubling of distance decay with some interference minima. Thus at least qualitatively the agreement appears good.

Appendix B contains a listing of the program for the Inversion case.

5.0 CONCLUSIONS

Approximate solutions of the Hankel transformed acoustic wave equation with a particular, realistic and well-developed vertical sound speed (or temperature) profile have been obtained for both the lapse and inversion cases. These solution are quite complex and exact inversion of the transformed solution does not appear possible. Both approximate inversion using contour integration and the saddle point approach and numerical inversion have been used to obtain the physical solution in the case of lapse conditions. Only numerical methods have been used with inversion condition.

The lapse case shows the expected behavior: an interference pattern with a 6 dB/doubling of distance decay within the shadow region; a rapid decrease in sound level in the vicinity of the geometric shadow boundary; and approximately a 6 dB/doubling of distance decay well within the shadow region. Similar behaviors occur for both inversion methods but the contour integration - saddle point method yields and larger decrease in the sound level on passing into the shadow than the numerical integration technique. The origin of this difference has not been determined. The contour integration - saddle point method results appears to agree with the empirical model of Weiner and Keast [17] better then the results of the numerical inversion technique. Since the techniques are applied to the same approximate solution of the transformed acoustic wave equation the difference must result from the inversion techniques. The numerical technique also produces a weak interference-like behavior far into the shadow region. This appears to be artifact of the numerical method as is the increase in sound level that frequently occurs as the maximum distance for the inversion technique is approached.

Agreement between the results and experimental data is fair within the shadow boundary. The level is predicted well but the location of interference maxima and minima are not accurately predicted. This may be due to the poor fit of the temperature profile to the measured profile. No data appears to be available that both gives a temperature profiles and sound levels in the shadow region.

The inversion case shows an interference pattern with a 6 dB/doubling of distance decay out to distances of the order of thirty meters for realistic temperature profiles. Beyond this distance the decay rate appears to remain nearly the same but the structure of minima and maxima becomes irregular. This tends to agree with a simple geometric argument since "trapped" rays start to reappear in a ray diagram at such distances. Little data is available for comparison in the inversion case.

6.0 REFERENCES

1. Best, A. C., "Transfer of Heat and Momentum in the Lowest Layers of the Atmosphere," Meterological Office Geophysical Memorandum No. 65, 1935.

2. Geiger, R., The Climate Near the Ground, Harvard Univ., Cambridge, 1965.

3. Reynolds, D. D., Engineering Principles of Acoustics, Allyn and Bacon, Boston, 1980.

4. Butterworth, J. R., "An Investigation of the Sound Pressure Levels in a Acoustic Shadow," M.S. Thesis, University of Utah, 1979.

5. Van Moorhem, W. K., Ma, Y., and Brown, J. M., "Ray Paths near the Ground in a Realistic Thermally Stratified Atmosphere," J. Acoustical Society of America, **80**, 650, 1986.

6. Bracewell, R. N., The Fourier Transform and its Applications, McGraw-Hill, New York, 1986.

7. Nayfeh, A., Perturbation Methods, Wiley, New York, 1973.

8. Tables of the Modified Hankel Function of order One-Third and of their Derivatives, Harvard Univ., Cambridge, 1945.

9. Van Moorhem, W. K. and Landheim, G. K., "The Propagation of Plane Waves in a Thermally Stratified Atmosphere," J. Acoustical Society of America, **76**, 867, 1984.

10. Churchill, R. V., Complex Variables and Applications, McGraw-Hill, New York, 1960.

11. Mathews, J. and Walker, R. L., Mathematical Methods of Physics,

Benjamin Cummings, Menlo Park, CA, 1964.

12. Ma, Y. "Acoustic Wave Propagation Along a Finite Impedance Ground Under Temperature Lapse Conditions," M.S. Thesis, University of Utah, 1984.

13. Cheng, A. S. C., "Sound Propagation from a Point Source in a Thermally Stratified Medium," M.S. Thesis, University of Utah, 1985.

14. Richards, T. L. and Attenborough K., "Accurate FFT-Based Hankel Transforms for Prediction of Outdoor Sound Propagation," J. Sound and Vibration, **109**, 157, 1986.

15. Lloyd, S. R., "Nonisothermal Acoustic Propagation Analysis using Fast Fourier Transforms," M.S. Thesis, University of Utah, 1985.

16. Sutherland, L. C., and Brown, R., "Static Tests of Excess Ground Attenuation at Wallops Flight Center," NASA Contractors Report 3435, 1981.

17. Weiner, R. M., and Keast, D. N., "Experimental Study of The Propagation of Sound over Ground," J. Acoustical Society of America, **31**, 724, 1959.

18. Van Moorhem, W. K., "Sixth Semi-Annual Report to the National Aeronautics and Space Administration on Grant NAG-1-283 Acoustic Propagation in a Thermally Stratified Atmosphere," UTEC ME 86-034, Mechanical and Industrial Engineering Department, University of Utah, Salt Lake City, 1986.

7.0 LIST OF SYMBOLS

English

a	Sound speed.
A	Function defined by (2.4) or constant in (3.7).
B	Function defined by (2.5) or constant in (3.7).
C	Function defined by (2.6).
$C_1, ...C_{17}$	Constants.
D	Function defined by (2.7).
E	Function defined by (2.11).
f	Arbitary function
F	Function defined by (2.10).
F_1	Function defined by (2.25).
F_2	Function defined by (2.26).
F_3	Function defined by (2.27).
g	Function defined by (3.8).
G	Hankel transform of $\overline{G}$.
$\overline{G}$	Acoustic pressure with time dependence seperated out, see (3.1)
h_1	Modified one-third order Hankel function of the first kind, see (3.12).
h_2	Modified one-third order Hankel function of the second kind, see (3.12).
i	$\sqrt{-1}$
I	Intergal defined by (3.41).
J_0	Zero order Bessel Function.
K	Function defined by (3.25).
$K_1, ... K_{17}$	Constants
p	Acoustic pressure.
q	Constant determining the strength of a point acoustic source.
r	Horizontal distance from the source.

r	Horizontal distance from the source.
R_o	Function defined by (3.26).
R_1	Function defined by (3.29).
R_2	Function defined by (4.11).
s	Height of the point source above the ground.
t	Time.
T	Temperature.
z	Height above the ground surface.
Z	Acoustic impedance of the ground surface.

Greek

α	Scale factor for temperature, see (1.2).
β	Hankel transform variable replacing r, see (3.2).
γ	Function defined by (2.17).
δ	Delta function.
η	Function defined by (3.10).
θ	Angle an acoustic ray make relative to horizontal.
Θ	Limiting ray angle, see (2.28).
λ	$a_\infty/(\omega\,\alpha)$.
ξ	Arbitrary arguement
ρ	Density of the air.
τ	Function defined by (3.27).
ϕ	Function defined by (2.12).
Φ	Function defined by (3.9).
Ψ	Function defined by (3.29).
ω	Circular frequency.

Combinations

ΔT	Change in Temperature between the ground surface and far above the ground.

Subscripts other than given above

i	1,2 or 3.
o	Evaluated at the ground, or a reference value.
tp	Evaluated at a ray turning point.
s	Evaluated at the source height s.
z	Derivative with respect to height (g_z or g_{zz}) or evaluated at the height z.
∞	Evaluated at infinite height.

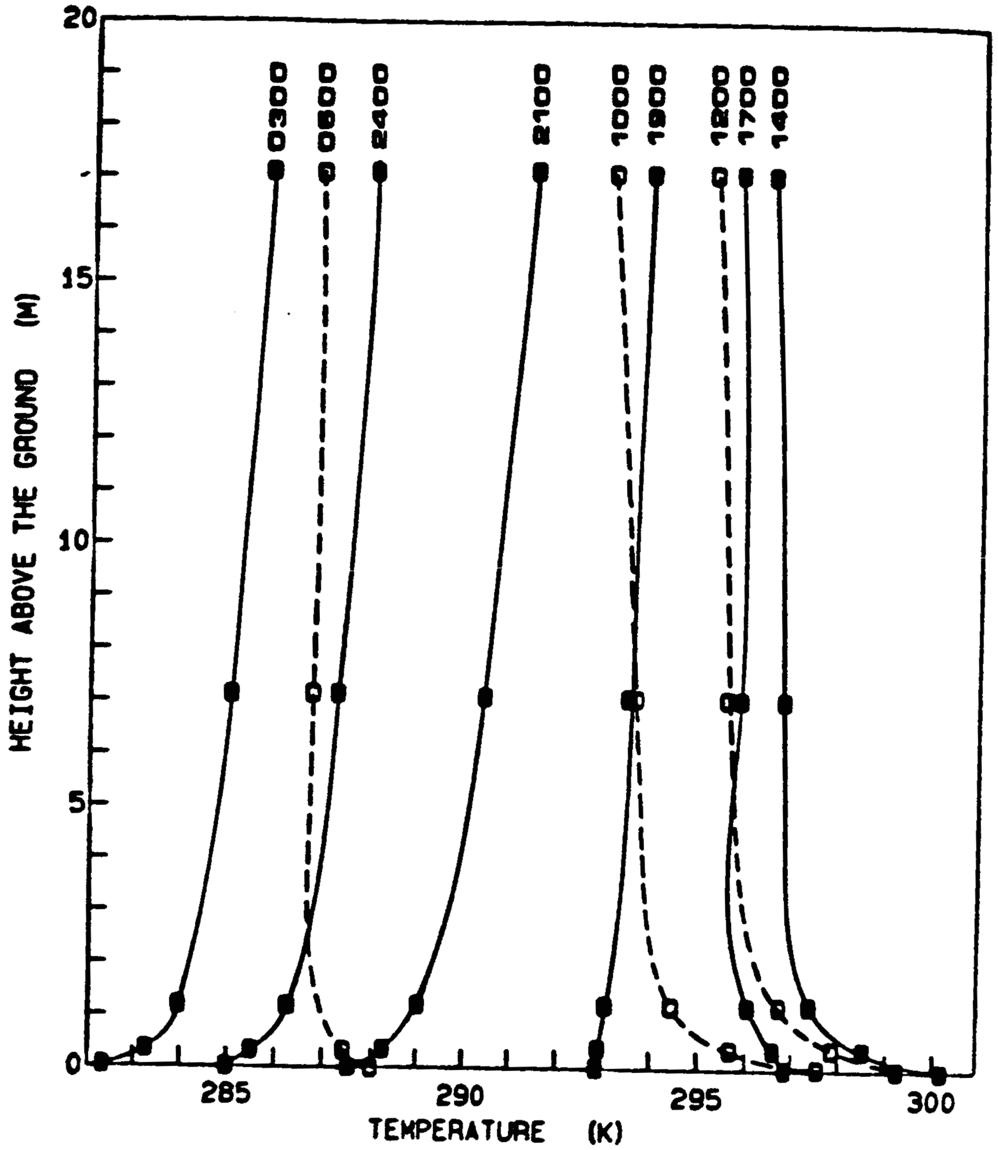


Figure 1.1. Temperature as a function of height above the ground for different times of the day as determined by Best [1].

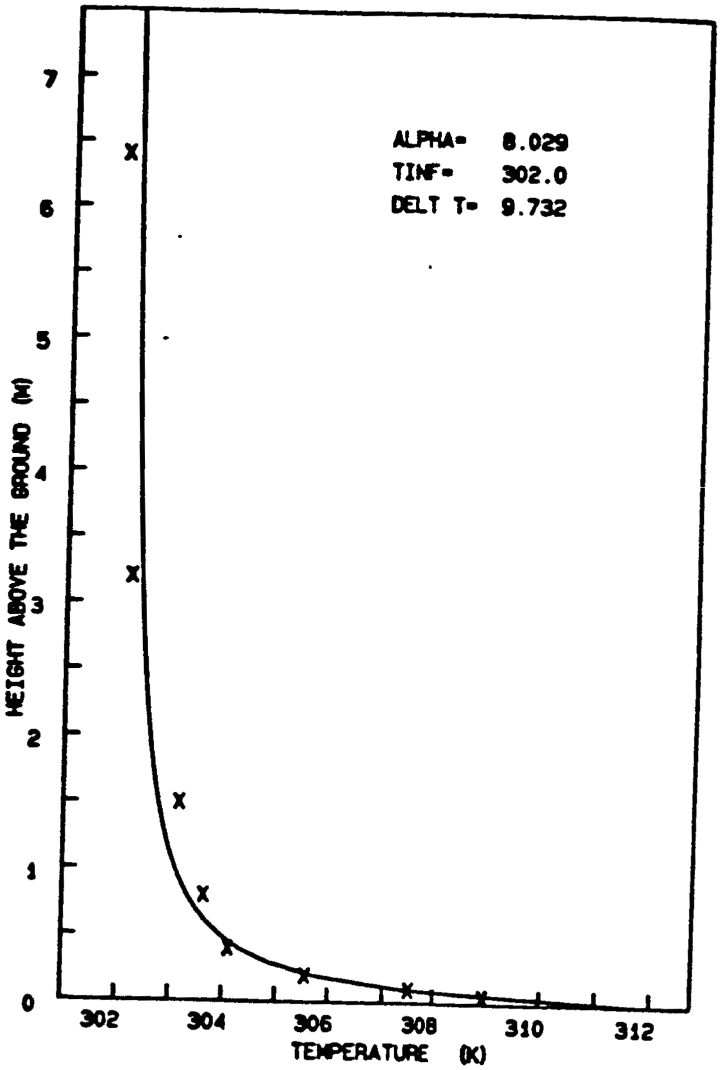


Figure 1.2. The present model of temperature as a function of height and a set of observations by Butterworth [4].

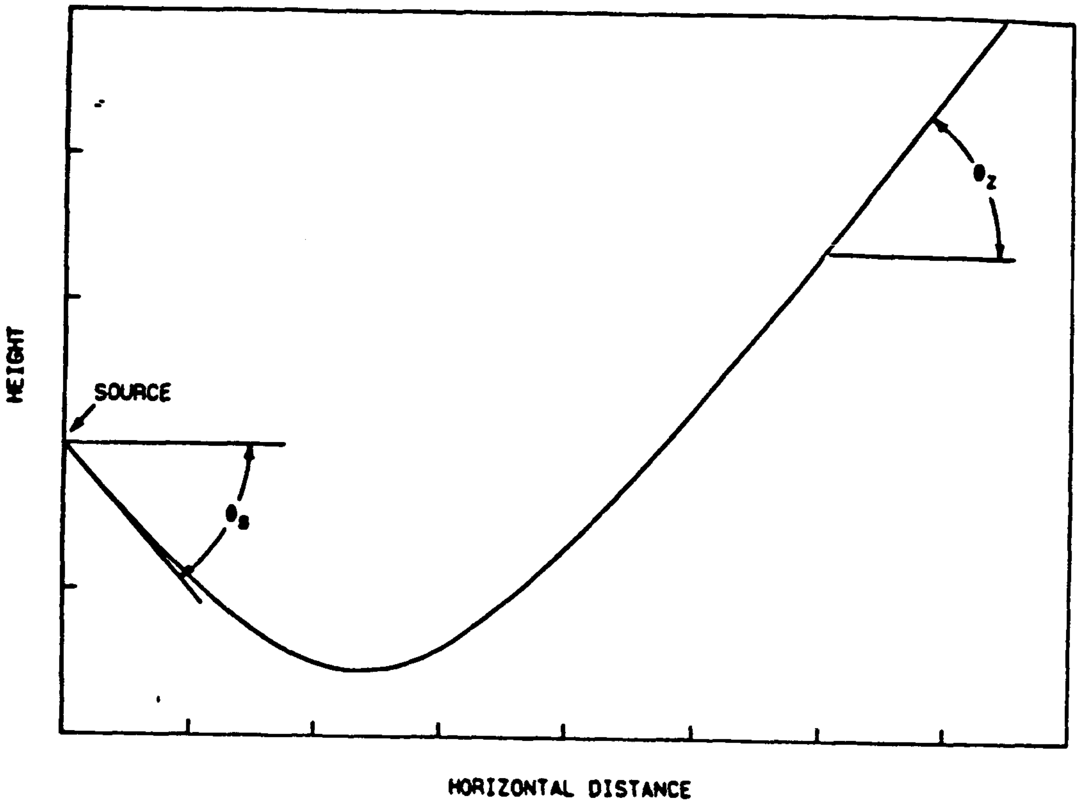


Figure 2.1. The nomenclature used in defining the rays.

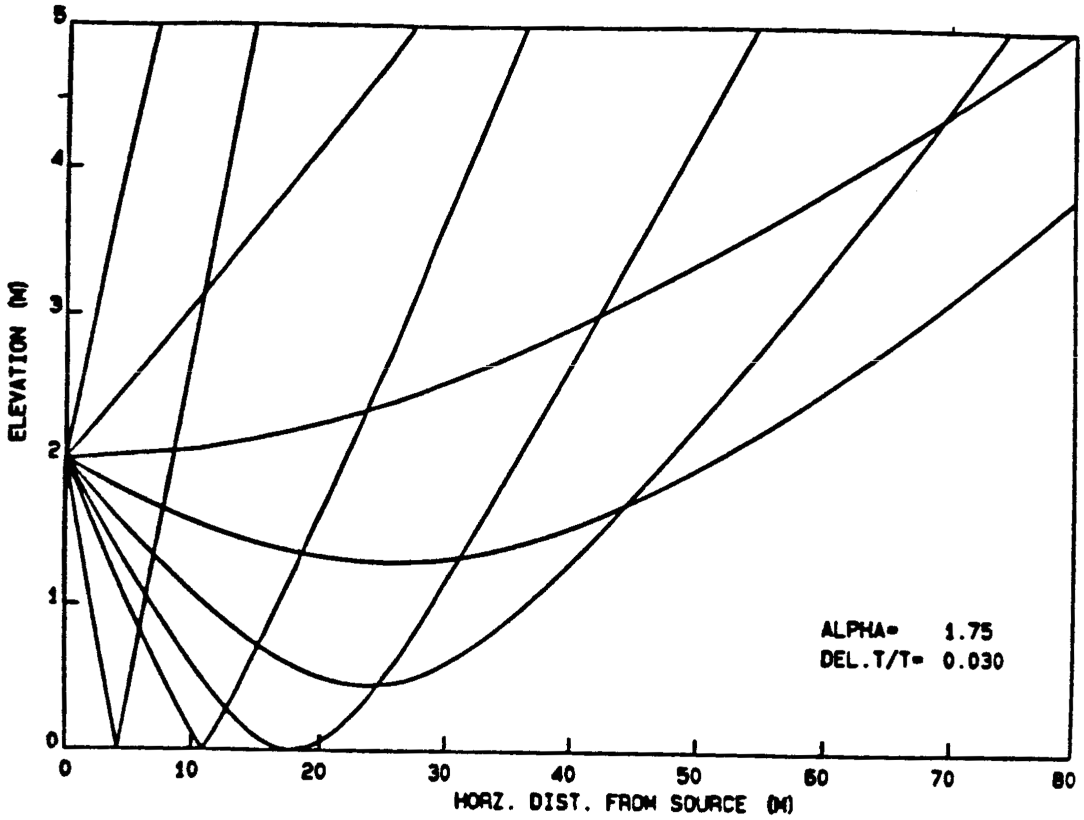


Figure 2.2. Acoustic rays for a lapse case with $\alpha = 1.75\ \mathrm{m}^{-1}$ and $\Delta T/T_{\infty} = 0.03$ with a source at a height of 2 m.

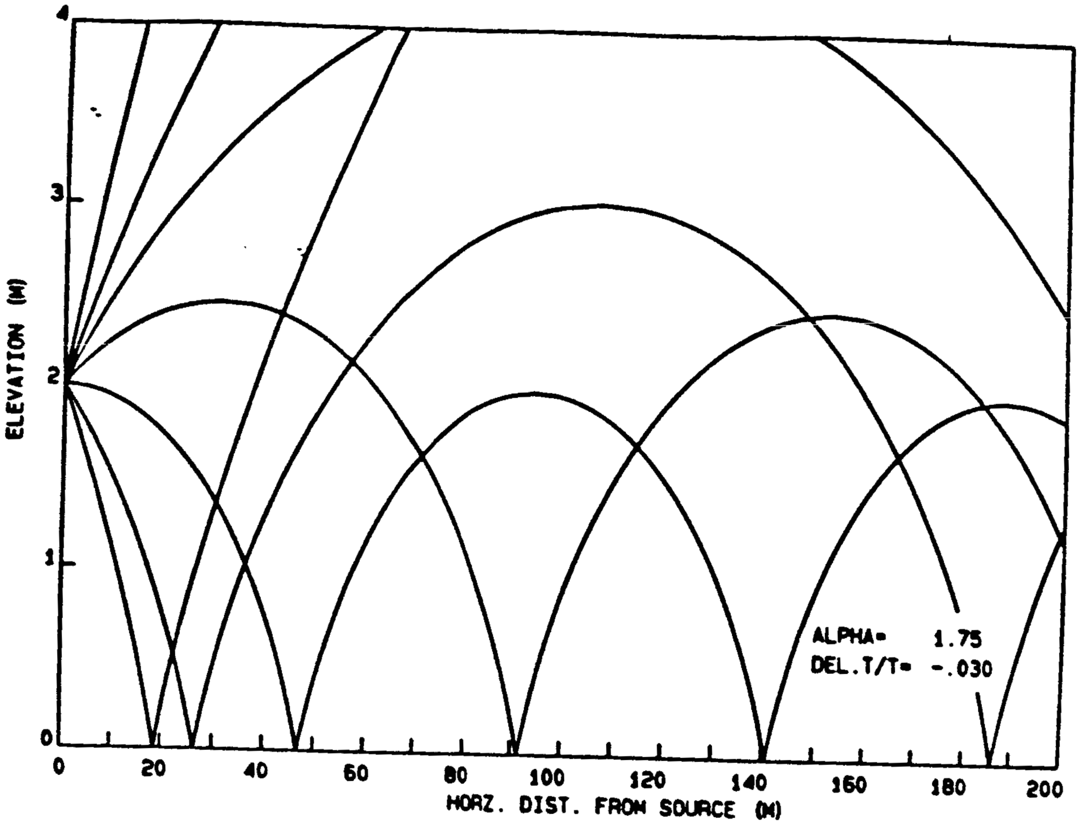


Figure 2.3. Acoustic rays for an inversion case with $\alpha = 1.75$ m^{-1} and $\Delta T/T_{\infty} = -0.03$ with a source at a height of 2 m.

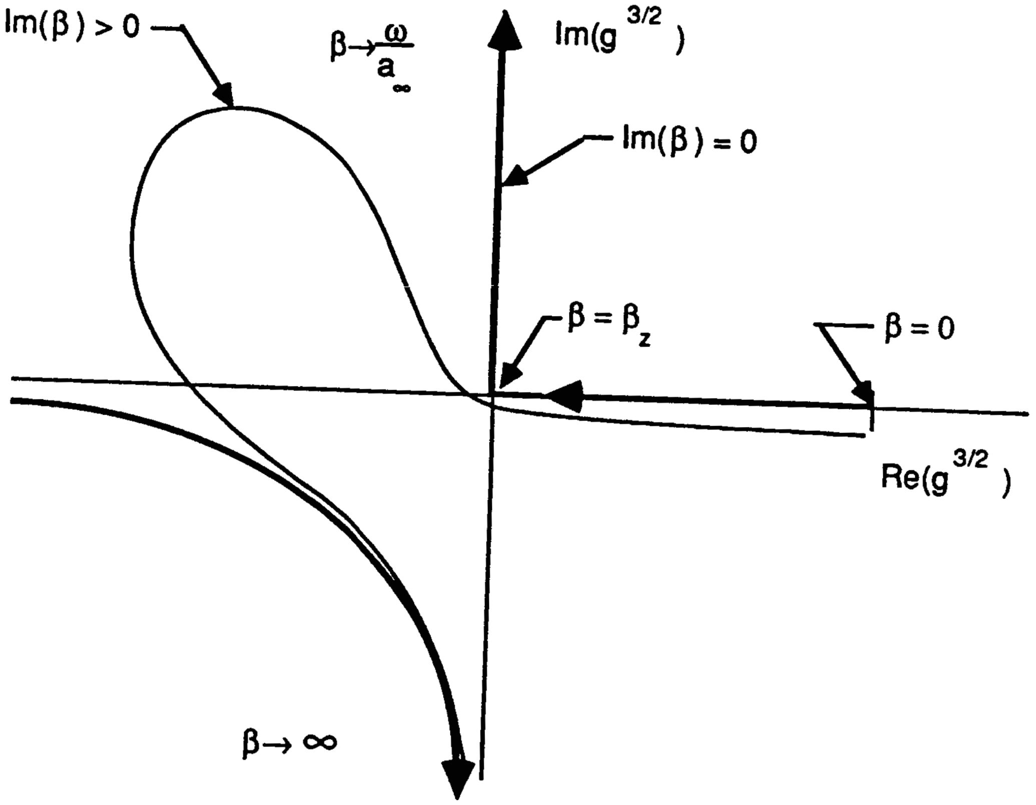
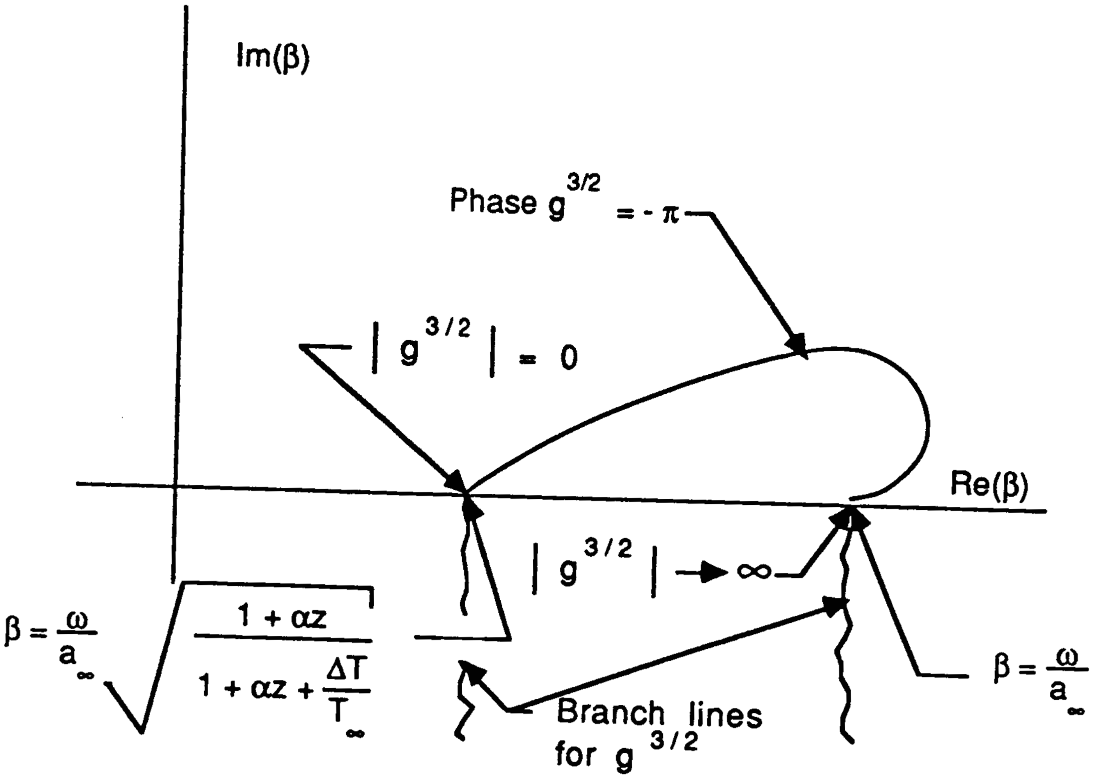


Figure 3.1. Path followed in complex $g^{3/2}$ - space as the real part of β varies from zero to infinity and the imaginary part of β is constant for the lapse case.



$\mathrm{Im}(\beta)$
Phase $g^{3/2} = -\pi$
$|g^{3/2}| = 0$
$\mathrm{Re}(\beta)$
$|g^{3/2}| \to \infty$
$\beta = \frac{\omega}{a_\infty}\sqrt{\frac{1+\alpha z}{1+\alpha z+\frac{\Delta T}{T_\infty}}}$
Branch lines for $g^{3/2}$
$\beta = \frac{\omega}{a_\infty}$

Figure 3.2. Sketch of the location of the branch line used for calculating $g\ (z,\beta) = (\ g^{3/2}\ (z,\beta)\ )^{2/3}$ in the lapse case.

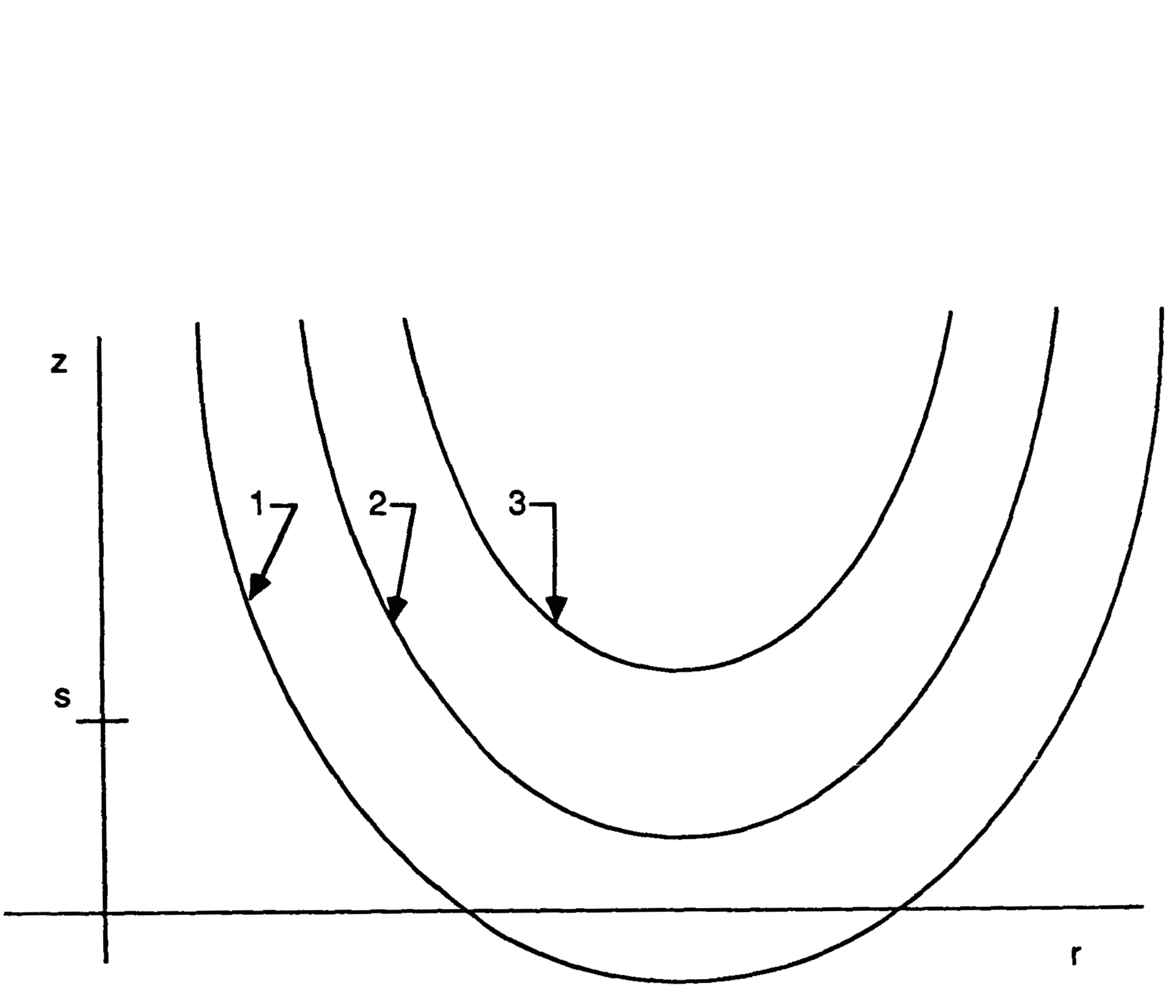


Figure 3.3. Sketch of the three types of physically occuring rays in the lapse case. Type 1 rays have their turning point below the ground surface. The turning point for type 2 rays is below the source and above the ground surface. Type 3 waves have a turning point above the source, this type of ray does not appear in a point source ray diagram but occurs in the superposition making up the inverse transform.

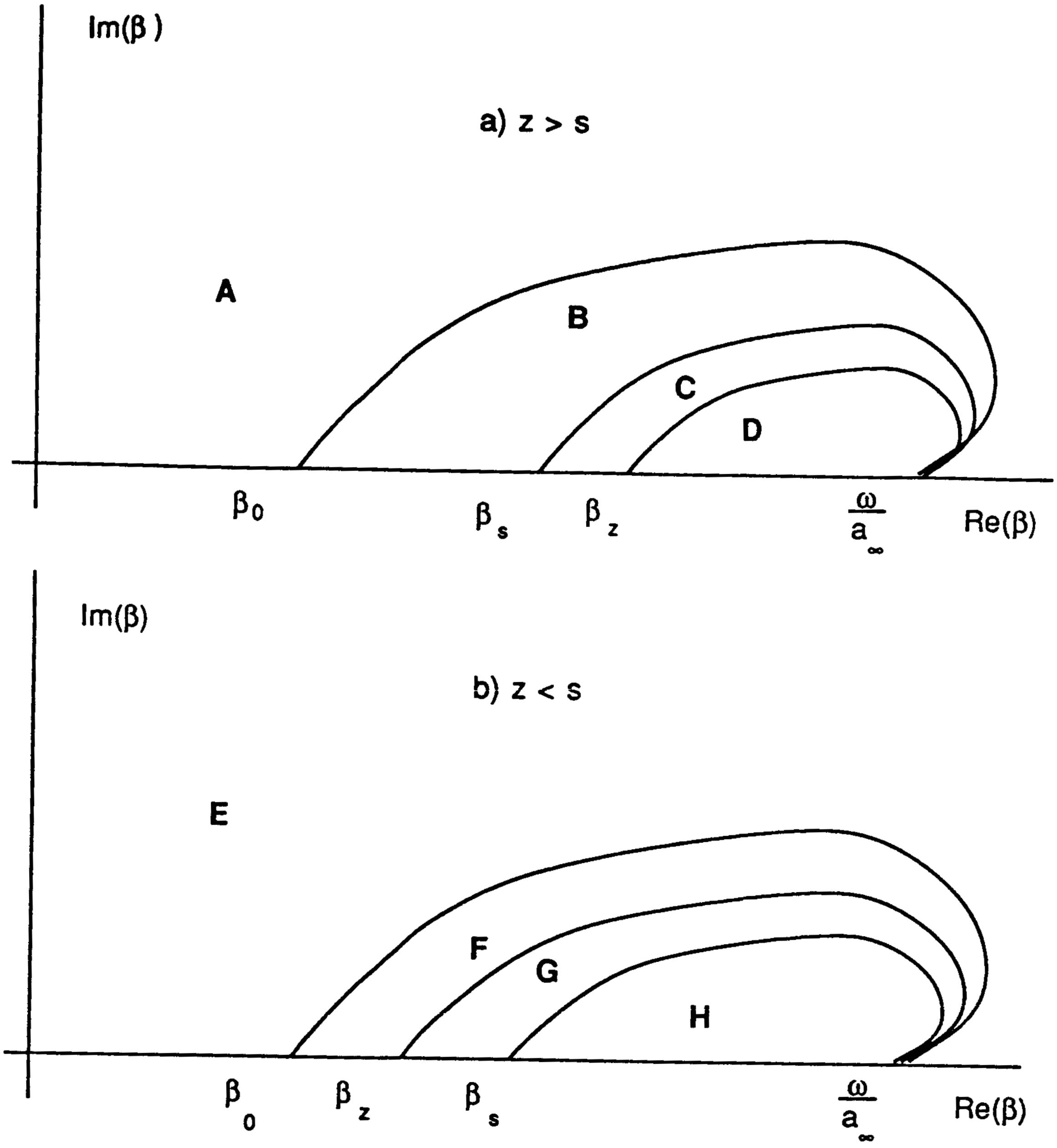


Figure 3.4. Sketchs of the regions in complex β-space where the various forms of the solution are valid for the lapse case. Part a) is for points above the source, part b) is for points below the source. The lines are branch lines for g (0,β), g (z,β) and g (s,β).

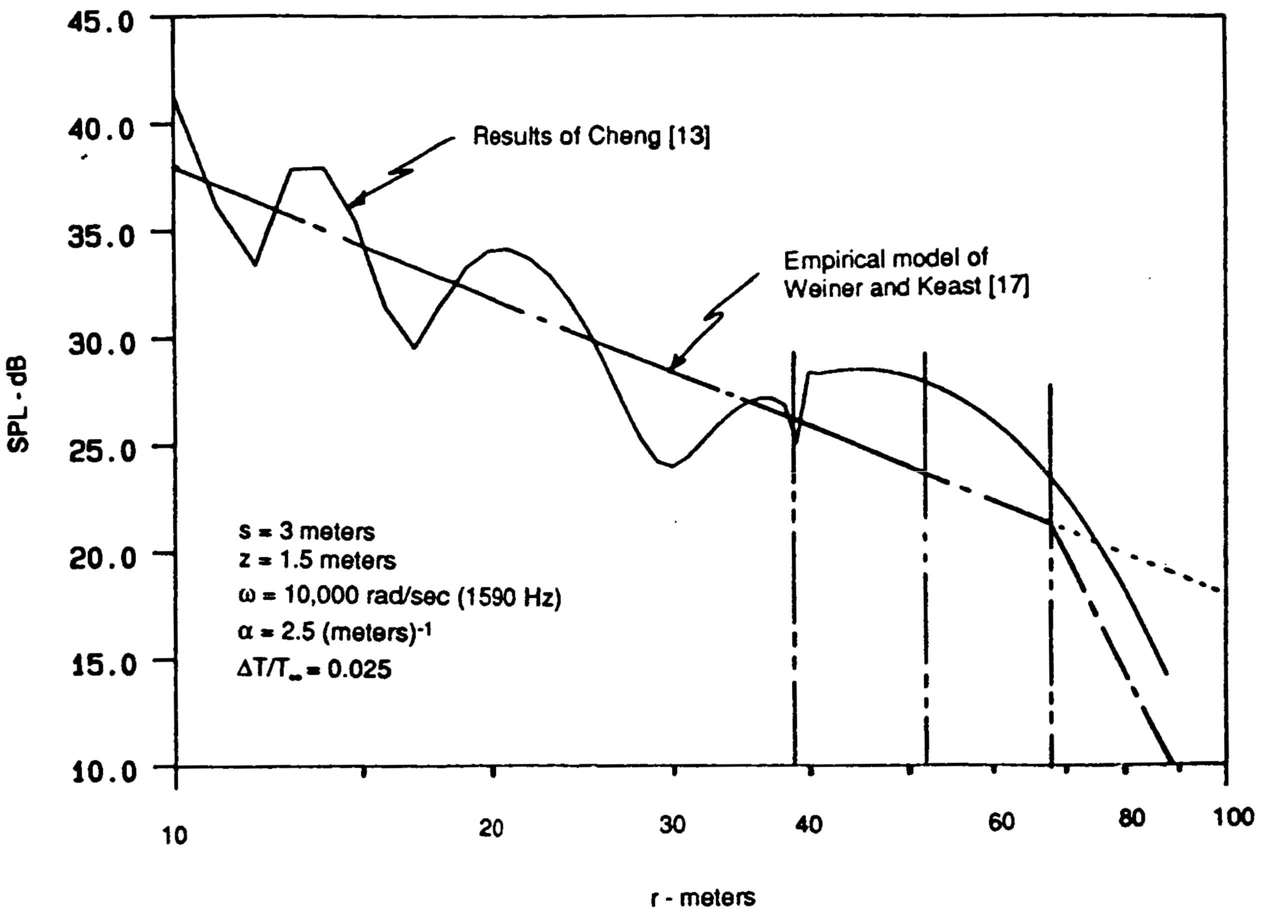


Figure 3.5. Typical results for a lapse case using the saddle point method only, from Cheng [13], as compared to the Weiner and Keast empirical model [17]. The solution extends only to the shadow boundary at about 68 meters.

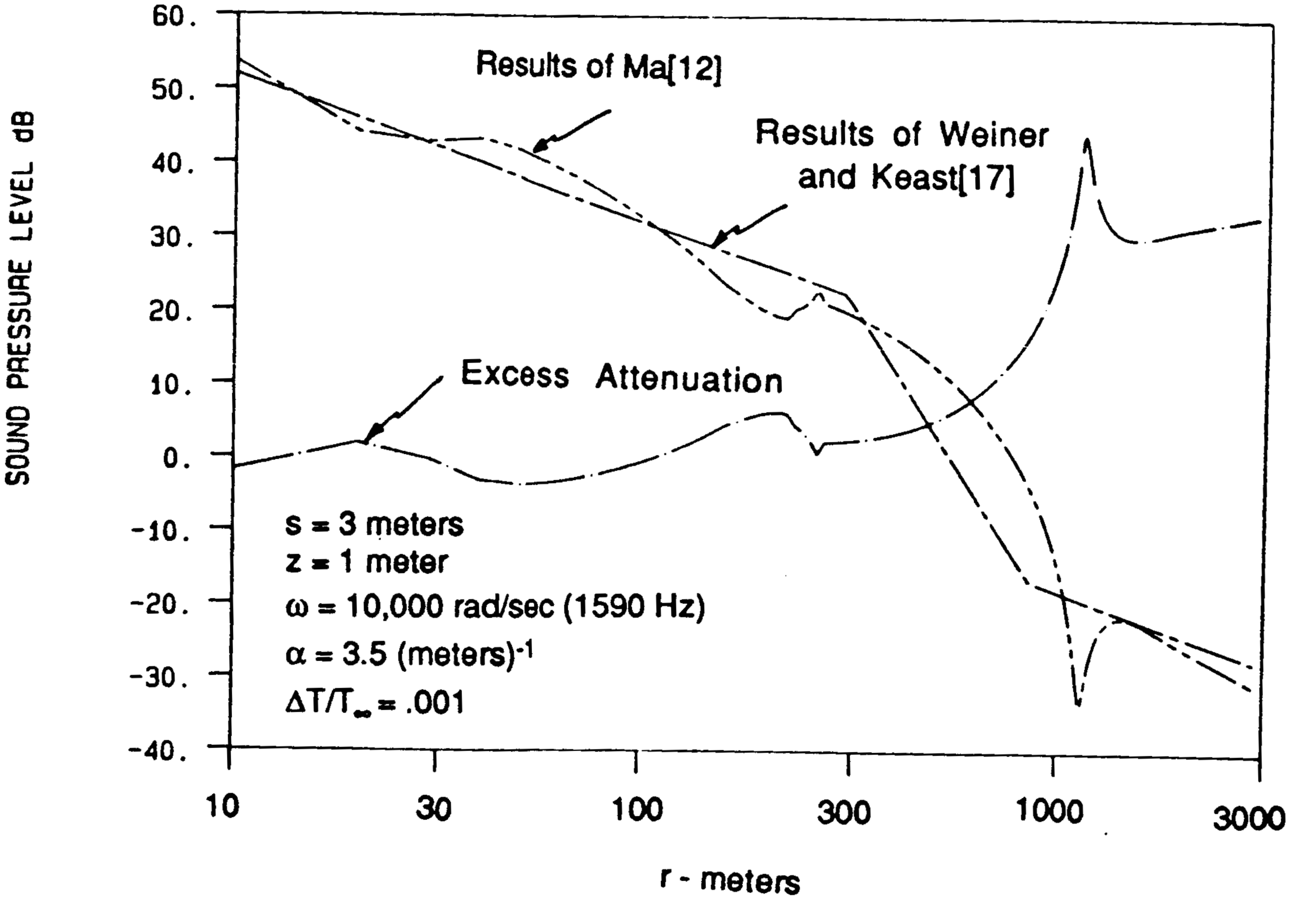


Figure 3.6. Typical results for a lapse case using the combined saddle point-contour integration method. From Ma [12].

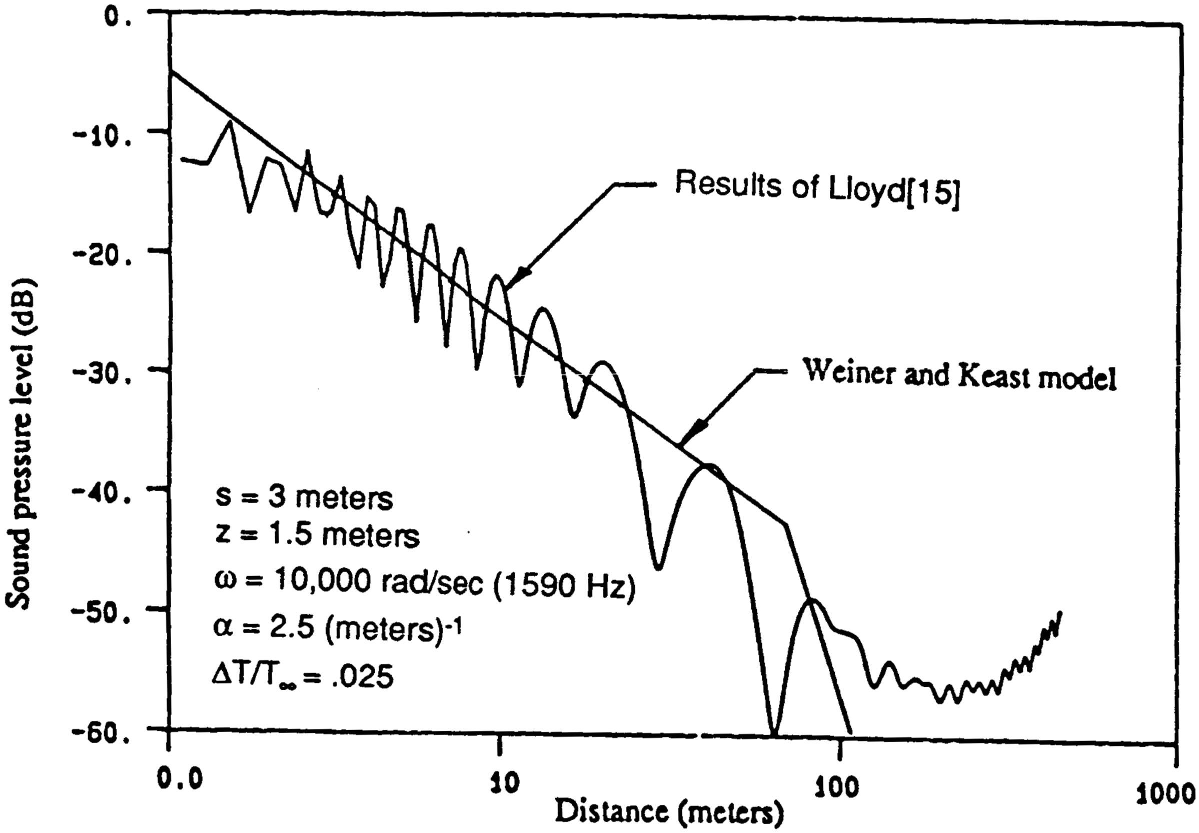


Figure 3.7. Typical results for a lapse case using the numerical inversion technique. From Lloyd [15].

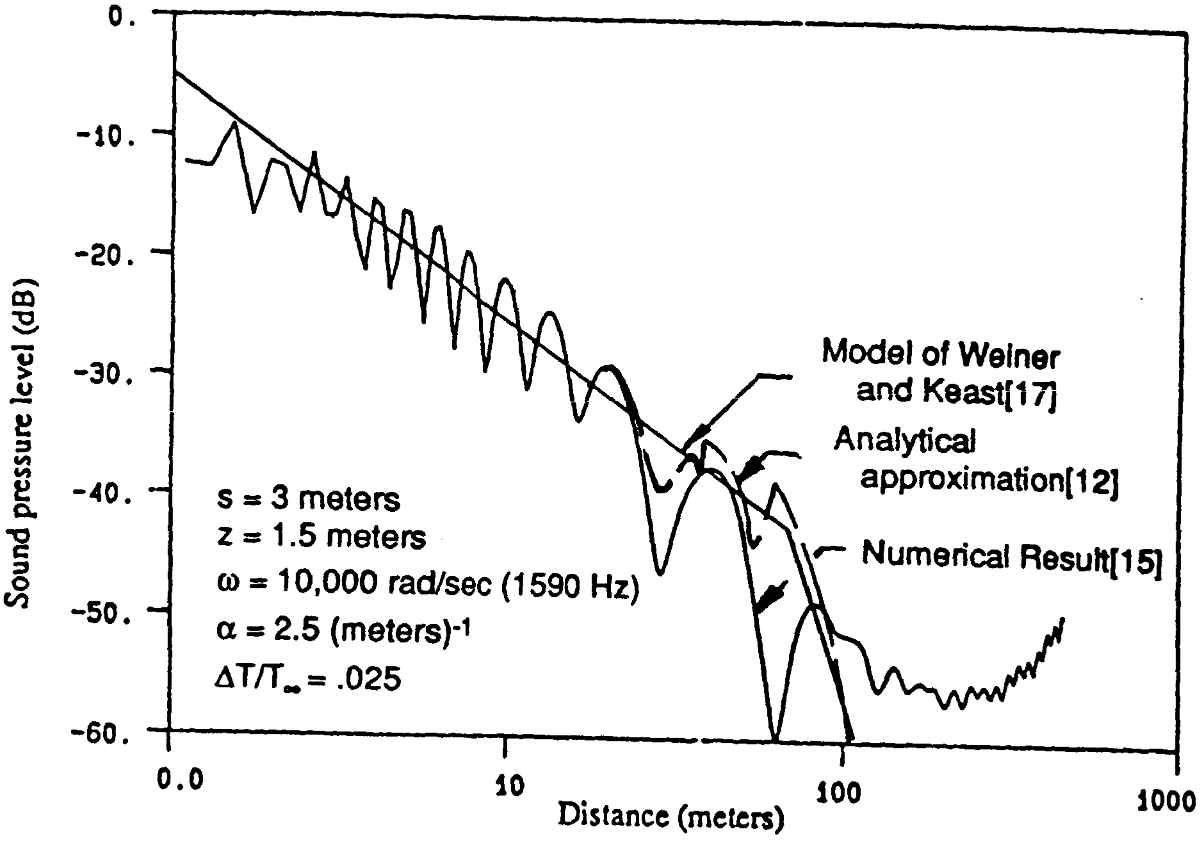


Figure 3.8. Comparison of the results of the saddle point-contour integration method and the numerical technique for a lapse case.

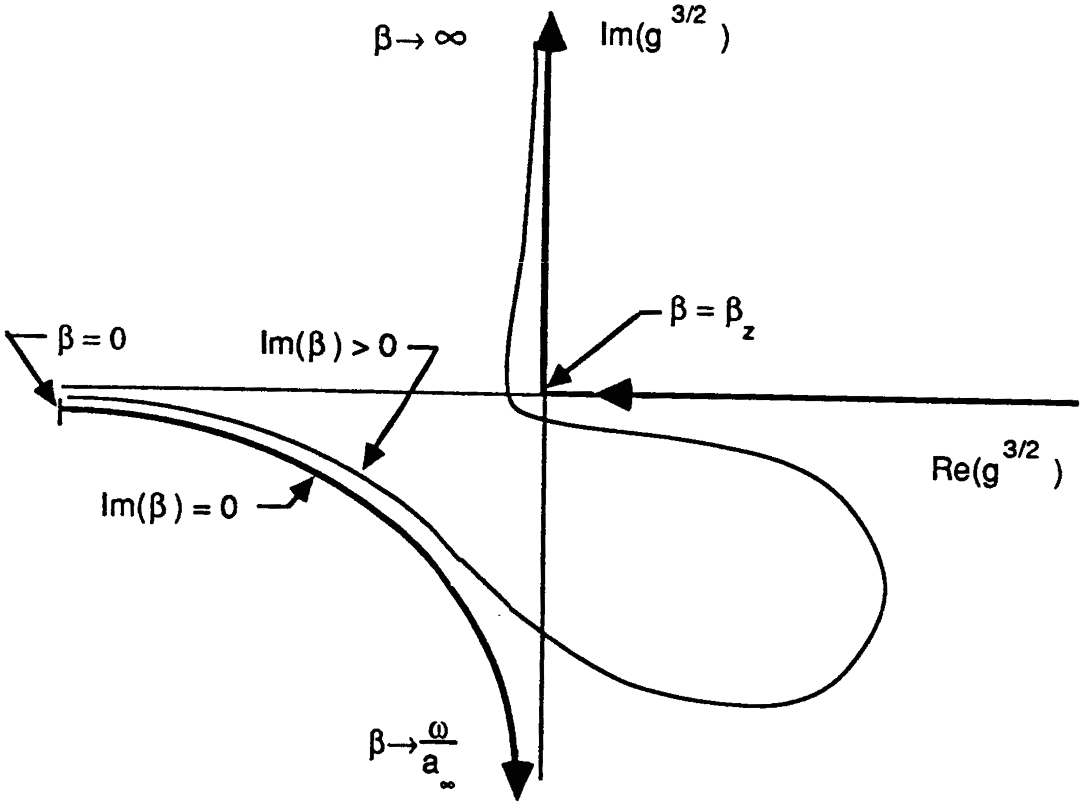


Figure 4.1. Path followed in complex $g^{3/2}$ - space as the real part of β varies from zero to infinity and the imaginary part of β is constant for the inversion case.

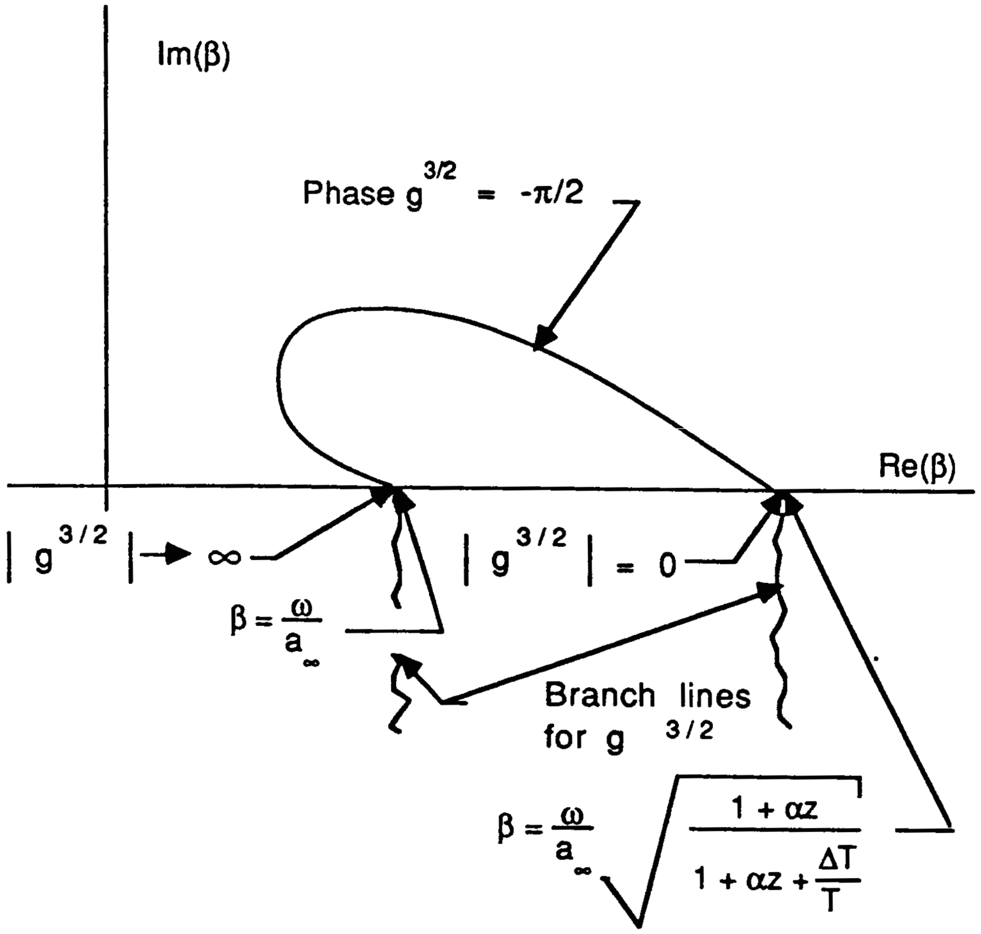

Im(β)
Phase $g^{3/2}$ = -π/2
Re(β)
$|g^{3/2}| \to \infty$
$|g^{3/2}| = 0$
$\beta = \frac{\omega}{a_\infty}$
Branch lines for $g^{3/2}$
$\beta = \frac{\omega}{a_\infty}\sqrt{\frac{1+\alpha z}{1+\alpha z+\frac{\Delta T}{T}}}$

Figure 4.2. Sketch of the location of the branch line for calculating g (z,β) = ($g^{3/2}$ (z,β))$^{2/3}$ in the inversion case.

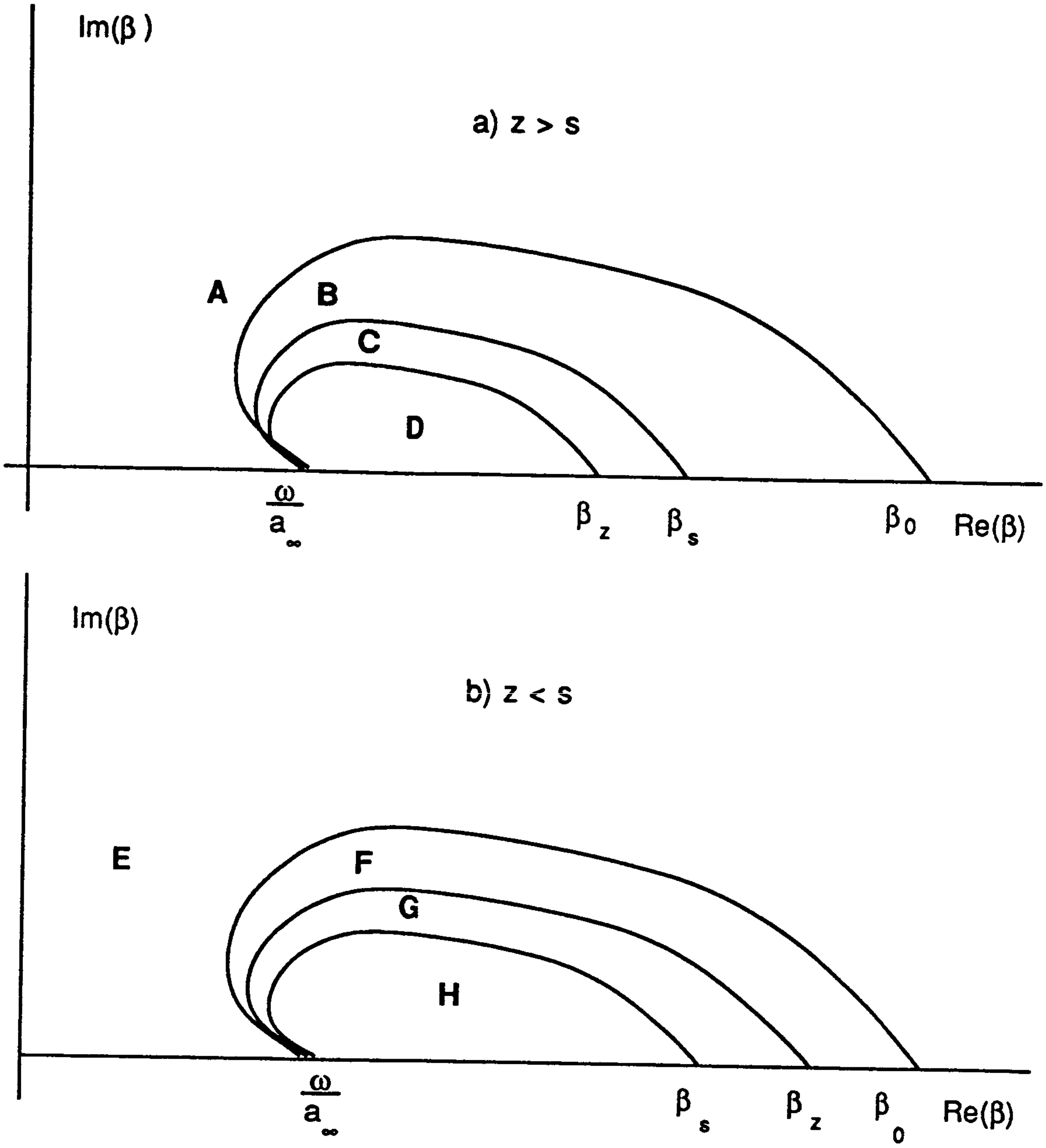


Figure 4.3. Sketch of the regions in complex β-space where the various forms of the solution are valid for the inversion case. The lines are branch lines for g (z,β), g (s,β) and g (0,β).

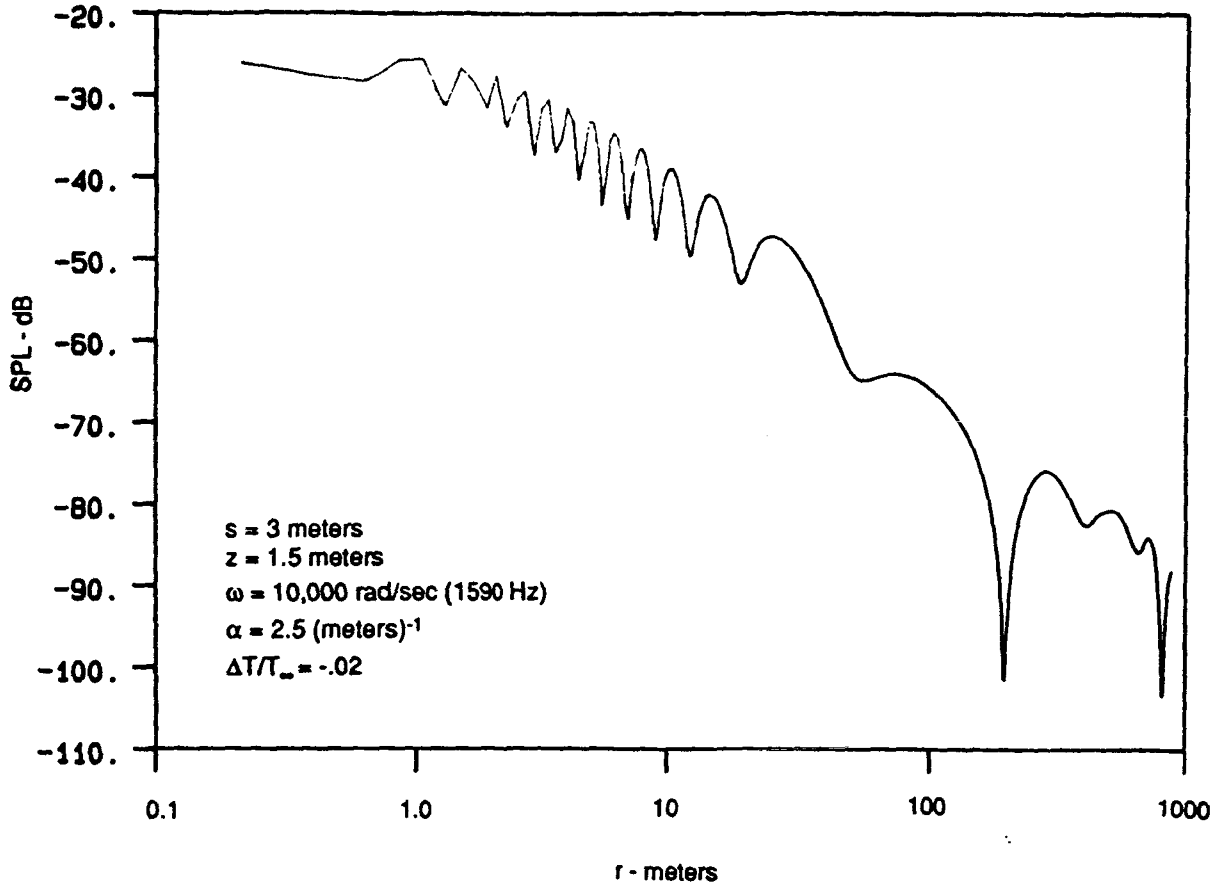


Figure 4.4. Typical results for an inversion case using the numerical inversion technique.

APPENDIX A - The Lapse Case Program

The material below is from Lloyd [15] and both describes the program used to calculate the transformed solution and to carry out the inversion and presents a listing of that program. The program is named FFTPRESS and was written to be used on a VAX 750. Descriptions are given both of the subroutines comprising the program, of the input variables and a set of "helpful hints" are given that may be useful in running the program. The intent of this section is not to describe in detail the operation of the program but to allow a somewhat experienced Fortran user to run the code as it was created.

SUBROUTINES

Input: Subroutine to input the necessary parameters to the main program. The following sentences summarize each of the input variables in the order they are requested. Tinf is the temperature at infinite height, normally 300 K. Tinf is used to calculate the speed of sound a. Dtot is the temperature change from infinity to the ground normalized by the temperature at infinity. Dtot is normally 0.025. Alpha is the term used in the temperature profile defining the altitude at which the temperature gradient becomes effective. Alpha is normally 2.5 (meters)$^{-1}$. Splref is the reference sound pressure level used in the calculations of sound pressure level in dB. Omega is the frequency of the sound source in rad/sec. Resistance is the flow resistance used in the Chessel model and is normally 300 cgs units. Zr is the height of the receiver in meters. Zs is the height of the source in meters. Alp is the amplitude of the imaginary component in transformed space. Alp greater than zero is integration above the real axis. The product of Alp and the step size ΔK should not exceed 0.01 and may be much smaller. Alp is replace by β in the thesis. Ns is used in an analytical function that is subtracted from the sampled solution to null the effects of off axis integration. Ns is normally 3. If Ns is equal to zero then no subtraction occurs. Ns is replaced by δ in the thesis. Me nonzero signals the inverse Hankel transofrm routine that the terms representing the integral extended to infinity are to be included in the inversion. Me is also the number of terms to be used and is normally 5. Me is replaced by M in the thesis. N1 is the number of points to be used. N1 depends on the maximum horizontal distance desired. N1 equal to 4096 points is a common value. Np and N1 are used interchangeably. N1 must be equal to an integer power of two. Delbeta or delK are the step size in complex K space. In the program Delbeta and DelK are used interchangeably. Delbeta also depends on the maximum horizontal distance desired and also on the maximum Beta allowed. This maximum Beta is very near to omega divided by the speed of sound. Beta and K are used interchangeably in the program and thesis.

Region: Subroutine used to determine which of 8 diffferent forms of the general solution are to be used. The selection depends on how the waves are interferring at that particular value of K. Region calls to subroutine g32all to identify if the complex part of $g^{3/2}$ has changed sign. The sign change indicates a different set of rays are combining to form the solution. With each set of rays a different form of the solution is required.

Hall: Subroutine to calculate the Hankel functions H_1 and H_2 and their derivatives in terms of the Airy functions, AI and BI. Hall calls to Cgbair to get the Airy functions needed.

Hall2: Subroutine to calculate only the Hankel functions.

Cgbair: Subroutine to calculate the Airy functions. Cgbair uses either an asymptotic or a small argument approximation of AI and BI depending on the value ofcomplex K.

Gzalll: Subroutine to calculate the derivatives of the $g^{3/2}$ function. These values are used to compute the reflection coefficients.

Gall: Calculates the g function needed to calculate the $g^{3/2}$ function.

Dafb2: Subroutine modified from Attenborough and Richards to calculate the inverse Hankel transform using fast Fourier transforms. Dafb2 calls to subroutine Zeta to calculate the terms representing the extension of the integral to infinity and to subroutine Fork used to perform the actual fast Fourier transform.

Zeta: Subroutine to compute the value of the integral to infinity.

Fork: An efficient fast Fourier transform taken from Attenborough and Richards program.

VARIABLES

Beta: Used interchangeably with K, both are the complex argument of the transformed solution.

Tau: The term τ used in the reflection coefficients developed by Van Moorhem.

Sci: The term ψ used in the reflection coefficient developed by Van Moorhem.

R_0: The actual reflection coefficient.

R_1: The modified reflection coefficient representing refraction.

En: The complex argument of the Hankel function. En is a function of K (or Beta) and the height of the source or receiver, whichever applies.

Rlemda: The ratio of omega to speed times alpha. The reciprocal of the wave number.

Zimped: The complex impedance of the ground normalized by the speed of sound and the density.

Z2 and Z3: Heights of the receiver and source, respectively.

Gbar (K) or Gbar(Beta): The sampled function to be inverted.

Gbar (r): The inverted solution. The real space answer.

G: The sound pressure level result.

Rad: The horizontal distance of the present (Gbar (r)

Rad2: The logarithm of Rad.

HELPFUL NOTES

1. All units used here are examples only. The only requirement in operation is to use a consistent set of units.

2. The output of sound pressure level and horizontal distance along with an echo of input parameters is written to file FOR044.dat in the present diretory.

3. To plot the result at the University of Utah Mechanical Engineering Vax system type RUN PLOTPROMPT and the rest is interactive.

4. Basic instructions for use on the University of Utah Mechanical Engineering Vax system are:

a. Log on using normal sequency of user name and password.

b. Type @Q to link all necessary files together. Instead of combining all files into a large file several small trackable files are used for ease of editing.

c. Type RUN FFTPRESS to begin execution.

d. Input the variables as requested by subroutine Input.

e. At completion FFTPRESS will display FINALLY FINISHED. To plot the results type RUN PLOTPROMPT. This is a standard plotting program that uses the system subroutine Mgraph. The data file name is FOR044.DAT. The data file contains 2 columns. The fist column of the data is the logarithm of the horizontal distance. The second column of the data is the sound pressure level. 15 lines are used at the beginning of FOR044.DAT to echo the input parameters, therefore tell PLOTPRMPT to begin accepting data at line sixteen of file FOR044.DAT. The plot will display on graphics terminals only. Mgraph asks if a hard copy is desired when crt plotting is finished. PLOTPROMT has autoscaling capability that can be turned on or off and offers many other self instructing options. Mgraph creates files named HPPLOT.HPL, however, it is recommended to change the name as soon as possible to avoid deletion of previous plots. If a hard copy plot is desired after exiting PLOTPROMT type PLOT then the file name.

f. Log off with command LO.

```
      PROGRAM FFTPRESS
C *****************************************************************
C  MAIN ROUTINE TO DEVELOPE THE GBAR(b,Z) TO BE INVERTED          *
C  THE MAIN WILL INPUT THE NECESSARY CONSTANTS,CALCULATE BETA,    *
C  SELECT FORM OF SOLUTION CALL THE SUBROUTINES AND COMPUTE RO,TAU, *
C  SCI,g,THE HANKEL FUNCTIONS AND FINALLY CALCULATE GBAR(b,Z).    *
C *****************************************************************
C  SUBROUTINES:
C       G32ALL:       FINDS THE FUNCTION g^3/2(b,z) USED TO DETERMINE
C                      WHICH REGION OF SPACE PRESENT BETA IS IN.
C       REGION:       GIVEN g^3/2 DETERMINES WHICH REGION OF
C                      SPACE PRESENT BETA IS IN.
C       GZALL:        FINDS gz(b,Z)=dg/dZ AND THE OTHER DERIVATIVES
C                      USED TO FIND K,TAU,SCI. ALSO RETURNS ETA(b,Z)
C                      THAT IS USED AS THE ARGUMENT FOR HANKEL
C                      FUNCTIONS. GZALL CALLS TO GALL TO GET g.
C       GALL:         CALCULATES g FUNCTION.
C
C       HALL:         CALCULATES THE HANKEL FUNCTIONS FROM AIRY
C                      FUNCTIONS. CALLS CGBAIR TO GET AIRY FUNCTIONS
C       HALL2:        CALCULATES ONLY THE HANKEL FUNCTIONS NOT THE
C                      DERIVATIVES AS HALL DOES. CALLS CGBAIR ALSO.
C       DAFB2:        GIVEN GBAR(b,Z) USES METHOD DEVELOPED BY
C                      RICHARDS AND ATTENBOROUGH TO PERFORM THE
C                      HANKEL INVERSION. USES SUBROUTINES ZETA,FORK.
C                      DAFB2 MAKES SEVERAL CORRECTIONS TO A GENERAL
C                      FFT. THE STANDARD CODE IS TAKEN FROM RICHARDS
C                      AND ATTENBOROUGH PROGRAM.
C       ZETA:         FORMS THE SUM OF ME TERMS WHICH APPROXIMATES
C                      GBAR(b,Z) TO INFINITY IN THE BETA SPACE.
C       FORK:         A VERY FAST FFT USED TO PERFORM ACTUAL
C                      INVERSION OF GBAR(b,Z) FROM THE BETA SPACE.
C  INPUTS:
C       TINF:         THE TEMPERATURE AT VERY LARGE Z
C       DTOT:         THE DELTA T/T PARAMETER REPRESENTING THE
C                      TEMPERATURE GRADIENT.
C       ALPHA:        PARAMETER USED IN DEFINITION OF TEMPERATURE
C                      GRADIENT.
C       SPEED:        SPEED OF SOUND AT T(INF).
C       OMEGA:        FREQUENCY OF SOUND IN RADIANS/SEC.
C       Z:            FIXED DISTANCE TO THE OBSERVER Z2 IN PROGRAM.
C       S:            FIXED DISTANCE TO THE SOURCE   Z3 IN PROGRAM.
C       SPLREF:       REFERENCE SOUND PRESSURE LEVEL USED TO COMPUTE
C                     THE SOURCE STRENGH Q.
C       RESISTANCE:   GROUND RESISTANCE IN THE CHESSELL MODEL.
C       ALP:          THE TERM USED TO INTEGRATE OFF REAL BETA AXIS.
C       NS:           PARAMETER IN THE ANALYTICAL FUNCTION IN DAFB2.
C       ME:           PARAMETER TO PRODUCE SUM TO INFINITY IN DAFB2.
C       N1:           SIZE OF ARRAY TO BE INVERTED.
C       DELBETA:      STEP SIZE USED FOR BETA ALSO DELK IN DAFB2.
C
C  NOTE:     K AND BETA AND DELK AND DELBETA ARE USED INTERCHANGEABLY
```

```
C
C  VARIABLES:
C       BETA:       INDEPENDENT VARIABLE IN TRANSFORMED SPACE.
C       TAU:        TAU DEFINED IN PAPER BY VAN MOORHEM.
C       SCI:        SCI DEFINED IN PAPER BY VAN MOORHEM.
C       RO:         RO  CONSTANT DEFINED IN VAN MOORHEM.
C       R1:         R1  CONSTANT DEFINED IN VAN MOORHEM.
C       EN:         HANKEL FUNCTION ARGUMENT.
C       RLEMDA:     OMEGA/(ALPHA*SPEED).
C       BRO:        BRANCH CUTS IN THE BETA SPACE ASSOCIATED WITH
C       BRZ:          THE SQUARE ROOT AND 3/2 POWER FUNCTIONS
C       BRS:          IN g AND g THREE HALF.
C       BRW:        BRANCH CUT AT OMEGA/SPEED.
C       ZIMPED:     GROUND IMPEDENCE NORMALIZED BY DENSITY AND SPEED.
C       GZ:         dg/dZ FROM GZALL1.
C       GZZ:        d2g/dZ2 FROM GZALL1.
C       OTHER DERIVATIVES PER THIS NOTATION
C       Z1:         REFERENCE DISTANCE 0.0.
C       Z2:         Z AS ABOVE.
C       Z3:         S AS ABOVE.
C ************************************************************
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      INTEGER IREGION
      COMMON /INTEG/ NS,ME,N1
      COMMON /AFB2IN/ ALP,DELBETA
      COMMON /CONSTANT1/ SPEED,OMEGA
      COMMON /CONSTANT2/ ALPHA,DTOT
      COMMON /CONSTANT3/RLEMDA,Q
      COMMON /CONSTANT4/CMPI,CK,RO,R1
      COMMON /HEIGHT/ Z1,Z2,Z3
      COMMON /CETA/Z1EN,Z2EN,Z3EN
      COMMON /BRANCH/BRO,BRS,BRZ,BRW
      COMPLEX*16 BETA,GZ,GZZ
      COMPLEX*16 H2,H21
      COMPLEX*16 EN,Z1EN,Z2EN,Z3EN
      COMPLEX*16 H1,H11
      COMPLEX*16 CK,TAU,SCI,ZIMPED
      COMPLEX*16 DUM1,DUM2,RO,R1,CMPI
      COMPLEX*16 GBAR(32768)
      CALL INPUT(TINF,SPLREF,RESISTANCE)
      Q=.00002*4.*3.1415926*4.67D0*(10.**(SPLREF/20.D0))
      SPEED=DSQRT(1.4D0*287.D0*TINF)
      PRINT *,'THE FOLLOWING IS AN ECHO OF THE INPUT '
      PRINT *,'IN THE FOLLOWING ORDER ALP DELTA ME NP DELBETA'
      PRINT *,'SPEED OMEGA ALPHA DTOT Z1 Z2 Z3 RESISTANCE'
      PRINT *,'TINF,SPLREF'
      PRINT *,' '                         ! SKIP A LINE
      PRINT *,'THESE VALUES ARE ALSO WRITTEN TO FILE 44'
      PRINT *,ALP,NS,ME,N1,DELBETA,SPEED,OMEGA,ALPHA,DTOT
      PRINT *,Z1,Z2,Z3,RESISTANCE,TINF,SPLREF
      WRITE(44,*) 'ECHO OF INPUT ALP NS ME DELBETA SPEED OMEGA ALPHA
   &            DTOT Z1 Z2 Z3 RESISTANCE TINF SPLREF'
```

```
      WRITE(44,*) ALP,NS,ME,N1,DELBETA,TINF,SPLREF,OMEGA,ALPHA,
     & DTOT,Z1,Z2,Z3,RESISTANCE,Q,SPEED
      RLEMDA=OMEGA/(SPEED*ALPHA)
      PI=3.1415926
      CMPI=(0.0,1.DO)
C  FUNCTIONS TO COMPUTE THE COMPLEX IMPEDENCE FROM CHESSELL MODEL
      FR=OMEGA/2.DO/PI
      RATIO=FR/RESISTANCE
      R=1.+9.08DO*RATIO**(-.75DO)
      X=-11.9DO*RATIO**(-.73DO)
      ZIMPED=DCMPLX(R,X)
      BRO=OMEGA/SPEED*DSQRT(1.DO/(1.DO + ALPHA*Z1+DTOT))
      BRZ=OMEGA/SPEED*DSQRT((1.DO+ALPHA*Z2)/(1.DO+ALPHA*Z2+DTOT))
      BRS=OMEGA/SPEED*DSQRT((1.DO+ALPHA*Z3)/(1.DO+ALPHA*Z3+DTOT))
      BRW=OMEGA/SPEED
      PRINT *,' '
      PRINT *,' '
      PRINT *, 'THE BRANCH CUTS ARE',BRO,BRZ,BRS,BRW
      WRITE(44,*) 'THE BRANCH CUTS ARE',BRO,BRZ,BRS,BRW
      DO 1,I=1,N1
       BETA=DCMPLX(DFLOAT(I-1),(-ALP))*(DELBETA)
       IF (ABS(BETA) .LE. .00000001) THEN
        BETA=DCMPLX(.0000001DO,(-ALP))*DELBETA
       END IF
       CALL GZALL1(Z1,BETA,GZ,GZZ,EN)
       Z1EN=EN
       DUM1=GZ
       DUM2=GZZ
       CALL GZALL1(Z2,BETA,GZ,GZZ,EN)
       Z2EN=EN
       CK=Q/12.DO/CMPI/(RLEMDA**(2.DO/3.DO))*1.0/(CDSQRT(GZ))
       CALL GZALL1(Z3,BETA,GZ,GZZ,EN)
       Z3EN=EN
       CK=CK*1.DO/(CDSQRT(GZ))
       TAU=ALPHA*RLEMDA-CMPI/2.DO*ZIMPED*DUM2/DUM1
       SCI=ZIMPED*((3./2.)**(2.DO/3.DO))*((RLEMDA)**(2.DO/3.DO))*DUM1
       CALL HALL(Z1EN,H2,H21,H1,H11)
       DUM1=TAU*H1+CMPI*SCI*H11
       DUM2=TAU*H2+CMPI*SCI*H21
       RO=-DUM1/DUM2
       DUM1=CMPI*PI/6.DO
       R1=(CDEXP(-DUM1)*(CMPI*RO))/((CDEXP(DUM1)**2.DO)+(RO**2.DO))
       CALL REGION(BETA,IREGION)
       GO TO(10,20,30,40,50,60,70,80),IREGION
C  **********  REGION 1 BEGINS HERE FOR Z2>Z3 OR Z>S  **********
  10   CONTINUE
        CALL HALL2(Z3EN,H2,H1)
        DUM1=H1+RO*H2
        CALL HALL2(Z2EN,H2,H1)
        GBAR(I)=CK*H2*DUM1
        GOTO 500
C  **********  REGION 2 BEGINS HERE FOR Z2>Z3 OR Z>S  **********
```

```
  20    CONTINUE
         CALL HALL2(Z3EN,H2,H1)
         DUM1=H1+R1*H2
         CALL HALL2(Z2EN,H2,H1)
         GBAR(I)=CK*H2*DUM1
         GOTO 500
C  **********  REGION 3 BEGINS HERE FOR Z2>Z3 OR Z>S  **********
  30    CONTINUE
         CALL HALL2(Z3EN,H2,H1)
         DUM1=(H1+RO*H2)*R1*CDEXP(CMPI*PI/3.DO)
         CALL HALL2(Z2EN,H2,H1)
         GBAR(I)=CK*H2*DUM1
         GOTO 500
C  **********  REGION 4 BEGINS HERE FOR Z2>Z3 OR Z>S  **********
  40    CONTINUE
         CALL HALL2(Z3EN,H2,H1)
         DUM1=CDEXP(CMPI*PI/3.DO)*R1*(H1+RO*H2)
         CALL HALL2(Z2EN,H2,H1)
         GBAR(I)=CK*DUM1*(H2+(CDEXP(CMPI*PI/3.DO)*H1))
         GOTO 500
C  **********  REGION 5 BEGINS HERE FOR Z2<Z3 OR Z<S  **********
   50  CONTINUE
        CALL HALL2(Z2EN,H2,H1)
        DUM1=H1+RO*H2
        CALL HALL2(Z3EN,H2,H1)
        GBAR(I)=CK*H2*DUM1
        GOTO 500
C  **********  REGION 6 BEGINS HERE FOR Z2<Z3 OR Z<S  **********
   60  CONTINUE
        CALL HALL2(Z2EN,H2,H1)
        DUM1=H1+R1*H2
        CALL HALL2(Z3EN,H2,H1)
        GBAR(I)=CK*H2*DUM1
        GOTO 500
C  **********  REGION 7 BEGINS HERE FOR Z2<Z3 OR Z<S  **********
   70  CONTINUE
        DUM1=CDEXP(CMPI*PI/3.DO)*R1
        CALL HALL2(Z2EN,H2,H1)
        DUM1=DUM1*(H1+RO*H2)
        CALL HALL2(Z3EN,H2,H1)
        GBAR(I)=CK*H2*DUM1
        GOTO 500
C  **********  REGION 8 BEGINS HERE FOR Z2<Z3 OR Z<S  **********
   80  CONTINUE
        DUM1=CDEXP(CMPI*PI/3.DO)*R1
        CALL HALL2(Z2EN,H2,H1)
        DUM1=DUM1*(H1+RO*H2)
        CALL HALL2(Z3EN,H2,H1)
        GBAR(I)=CK*(H1*CDEXP(CMPI*PI/3.DO)+H2)*DUM1
        GOTO 500
C  END OF GBAR(BETA) CALCULATIONS BASED ON REGIONS DETERMINED
C  MULTIPLY BY BETA ONLY TO MATCH VAN MOORHEM DEFINITION TO
```

```
C       RICHARDS AND ATTENBOROUGH.
  500   CONTINUE
        GBAR(I)=GBAR(I)*BETA
    1   CONTINUE
        PRINT *,' LAST BETA EXCUTED IS',BETA
        CALL DAFB2(GBAR)
        DO 2,I=1,N1/2
         RAD=2.DO*PI*DFLOAT(I-1)/(DFLOAT(N1)*DELBETA)
         IF(RAD.LE.0.0 ) THEN
          GOTO 5
         END IF
          RAD2=DLOG10(RAD)
          G=20.DO*DLOG10(ABS(GBAR(I)))
          WRITE(44,9093) RAD2,G
    5   CONTINUE
    2   CONTINUE
        PRINT *,'FINALLY FINISHED'
 9093   FORMAT(5X,3G18.8)
        STOP
        END
C
C
C
C
        SUBROUTINE INPUT(TINF,SPLREF,RESISTANCE)
C***********************************************************************
C THE PURPOSE OF THIS ROUTINE IS TO INPUT ALL NECESSARY PARAMETERS TO *
C  THE MAIN ROUTINE. THE DEFINITION OF EACH PARAMETER WILL BE DEFINED *
C  AT ITS RESPECTIVE INPUT.                                           *
C***********************************************************************
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        COMMON /INTEG/ NS,ME,N1
        COMMON /AFB2IN/ ALP,DELBETA
        COMMON /CONSTANT1/ SPEED,OMEGA
        COMMON /CONSTANT2/ ALPHA,DTOT
        COMMON /CONSTANT3/ RLEMDA,Q
        COMMON /CONSTANT4/ CMPI,CK,RO,R1
        COMMON /HEIGHT/ Z1,Z2,Z3
        COMMON /CETA/ Z1EN,Z2EN,Z3EN
        COMMON /BRANCH/ BRO,BRS,BRZ,BRW
        WRITE (6,899)
        WRITE(6,900)
        READ *, TINF
        WRITE (6,901)
        READ *, DTOT
        WRITE (6,902)
        READ *, ALPHA
        WRITE (6,903)
        READ *, SPLREF
        WRITE (6,904)
        READ *, OMEGA
```

```
      WRITE (6,905)
      READ *, RESISTANCE
      WRITE (6,906)
      READ *, Z2,Z3
      Z1=0.D0
      WRITE (6,899)
      WRITE (6,907)
      READ *, ALP
      ALP=-ALP        !  NECESSARY DUE TO PREVIOUS DEFINITION USED
      WRITE (6,908)
      READ *, NS
      WRITE (6,909)
      READ *, ME
      WRITE (6,910)
      READ *, N1
      WRITE (6,911)
      READ *, DELBETA
899   FORMAT('O',20X,1H ,///////////////)    ! CLEARS THE SCREEN
900   FORMAT('O',///,10X,38H THE FOLLOWING ARE PHYSICAL VARIABLES.
   &         ,//,12X,43H INPUT THE TEMPERATURE AT Z EQUAL INFINITY.
   &         ,/)
901   FORMAT (X,12X,45H INPUT THE TEMPERATURE CHANGE DELTA T/T(INF).
   &         ,/)
902   FORMAT (X,12X,46H INPUT THE TEMPERATURE PROFILE CONSTANT ALPHA.
   &         ,/)
903   FORMAT (X,12X,46H INPUT THE REFERENCE SOUND PRESSURE LEVEL.
   &         ,/)
904   FORMAT (X,12X,46H INPUT THE ANGULAR FREQUENCY IN RADIANS/SEC.
   &         ,/)
905   FORMAT (X,12X,47H INPUT THE CHESSELL MODEL FLOW RESISTANCE (cgs)
   &         ,/)
906   FORMAT (X,12X,46H INPUT RECEIVER AND SOURCE HEIGHTS SEPARATED
   &         ,/,25H                BY A COMMA.,/)
907   FORMAT (///,10X,37H THE FOLLOWING ARE NUMERIC VARIABLES.
   &         ,//,12X,39H INPUT ALP. THE IMAGINARY PART OF BETA.
   &         ,/,12X,43H NOTE POSITIVE VALUES ARE ABOVE THE AXIS.   ,/)
908   FORMAT (X,12X,46H INPUT THE DELTA PARAMETER USED IN INTEGRATION
   &         ,/)
909   FORMAT (X,12X,46H INPUT THE NUMBER OF TERMS IN SUM TO INFINITY.
   &         ,/)
910   FORMAT (X,12X,46H INPUT THE NUMBER OF POINTS TO BE USED.
   &         ,/)
911   FORMAT (X,12X,46H INPUT THE STEP SIZE IN BETA (OR K) SPACE.
   &         ,/)
      RETURN
      END
C
C
C
C
      SUBROUTINE REGION(BETA,IREGION)
C*********************************************************************
```

```
C  SUBROUTINE TO DETERMINE WHICH OF 8 REGIONS                        *
C   THE EVALUATION IS TO TAKE PLACE IN.                              *
C*********************************************************************

        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        INTEGER DOUBLE PRECISION K1
        COMPLEX*16 BETA,G32C1,G32C2,G32C3
        COMPLEX*16 G32PI,G32PI2,G32PI3
        COMMON /INTEG/NS,ME,N1
        COMMON /AFB2IN/ALP,DELBETA
        COMMON /CONSTANT1/SPEED,OMEGA
        COMMON /CONSTANT2/ALPHA,DTOT
        COMMON /CONSTANT3/RLEMDA,Q
        COMMON /CONSTANT4/CMPI,CK,RO,R1
        COMMON /HEIGHT/ Z1,Z2,Z3
        COMMON /CETA/Z1EN,Z2EN,Z3EN
        COMMON /BRANCH/BRO,BRS,BRZ,BRW
        INTEGER IREGION
         CALL G32ALL(Z1,BETA,G32C1)
         CALL G32ALL(Z2,BETA,G32C2)
         CALL G32ALL(Z3,BETA,G32C3)
C   IN THE EVENT THAT BRZ OR BRS AND BRW ARE VERY CLOSE THE FOLLOWING
C    IS USED TO INSURE PROPER REGION IS CHOSEN. W1 AND W2 ARE SIMPLE
C    WEIGHT FACTORS TO DETERMINE WHICH DIRECTION TO EVALUATE REGIONS.
C    BY WEIGHTING THE SELECTION DEPENDING HOW CLOSE BETA IS TO BRW
C    THE REGIONS ARE FOUND IN THE REVERSE ORDER.
         POSMAX=MAX(Z2,Z3)
        W1=1.0
        W2=1.0
        DUD=OMEGA/SPEED
      DUD=DUD*W1+DUD*W2*DSQRT((1.0+ALPHA*POSMAX)/(1+ALPHA*POSMAX+DTOT))
        DUD=DUD/2
         IF(Z2.LT.Z3) THEN
          GOTO 120
         END IF
C  FOR **(Z2>Z3 OR Z>S)** THE FOLLOWING IDENTIFY REGIONS OF SPACE
          AI1=DIMAG(G32C1)
          AI2=DIMAG(G32C3)
          AI3=DIMAG(G32C2)
        IF(REAL(BETA).GT.DUD) THEN
         GOTO 110
        END IF
         IF(AI3.GT.0.0) THEN
          IREGION=4
          GOTO 150
         END IF
         IF(AI2.GT.0.0) THEN
          IREGION=3
          GOTO 150
         END IF
         IF(AI1.GT.0.0) THEN
          IREGION=2
```

```
       GOTO 150
      END IF
       IREGION=1
       GOTO 150
 110  CONTINUE
      IF(AI1.LT.0.0) THEN
       IREGION=1
       GOTO 150
      END IF
      IF(AI2.LT.0.0) THEN
       IREGION=2
       GOTO 150
      END IF
      IF(AI3.LT.0.0) THEN
       IREGION=3
       GOTO 150
      END IF
      IREGION=4
      GOTO 150
C  FOR**(Z2<Z3 OR Z<S)** THE FOLLOWING IDENTIFY REGIONS OF SPACE
 120  CONTINUE
      AI1=DIMAG(G32C1)
      AI2=DIMAG(G32C2)
      AI3=DIMAG(G32C3)
      IF (REAL(BETA).GT.DUD) THEN
       GOTO 130
      END IF

      IF (AI3.GT.0.0) THEN
       IREGION=8
       GOTO 150
      END IF
      IF (AI2.GT.0.0) THEN
       IREGION=7
       GOTO 150
      END IF
      IF (AI1.GT.0.0) THEN
       IREGION=6
       GOTO 150
      END IF
      IREGION=5
      GOTO 150
 130  CONTINUE
      IF (AI1.LT.0.0) THEN
       IREGION=5
       GOTO 150
      END IF
      IF (AI2.LT.0.0) THEN
       IREGION=6
       GOTO 150
      END IF
      IF (AI3.LT.0.0) THEN
```

```
           IREGION=7
           GOTO 150
          END IF
          IREGION=8
          GOTO 150
  150      CONTINUE
          RETURN
          END
C
C
C
C
          SUBROUTINE G32ALL(Z,BETA,G32C)
C****************************************************************
C*                                                              *
C*              G32ALL CALCULATES g3/2 FUNCTION                 *
C*                                                              *
C*                                                              *
C****************************************************************
          IMPLICIT DOUBLE PRECISION  (A-H,O-Z)
          COMPLEX*16 BETA,PHI,SQRT1,SQRT2,FOO,G32C,MODLOG
          COMPLEX*16 S1,A,B,S
          COMMON /CONSTANT1/SPEED,OMEGA
          COMMON /CONSTANT2/ALPHA,DTOT
          COMMON /BRANCH/BRO,BRS,BRZ,BRW
          AA=1.+ALPHA*Z+DTOT
          B=1.-((SPEED*BETA/OMEGA)*(SPEED*BETA/OMEGA))
          S1=SQRT1(BETA,Z)
          S=SQRT2(BETA)
          PHI=S1/(S*DSQRT(AA))
          IF(DTOT.EQ..0.OR.CDABS(1.-PHI).LT.1D-8) THEN
                  FOO=.0
          ELSE
                  FOO=MODLOG((1.+PHI)/(1.-PHI))
          ENDIF
          G32C=DSQRT(AA)*S1-.5*DTOT*FOO/S
          RETURN
          END
C
C
C BEGIN OF FUNCTIONS USED ABOVE.
C
C ******************************************************************
C SQRT1 AND SQRT2 ARE FUNCTIONS TO CALCULATE SQRT(BETA) GIVEN       *
C  DESIRED BRANCH CUTS AND DIRECTION +IMAGINARY OR -IMAGINARY.      *
C  *****************************************************************
        FUNCTION SQRT1(BETA,Z)
        IMPLICIT DOUBLE PRECISION  (A-H,O-Z)
        COMPLEX*16 BETA,AA,SQRT1
        COMMON /CONSTANT1/SPEED,OMEGA
        COMMON /CONSTANT2/ALPHA,DTOT
        AA=1.-(((SPEED/OMEGA)*BETA)*((SPEED/OMEGA)*BETA))
```

```
      BB=1.+ALPHA*Z+DTOT
      BRAN1=(OMEGA/SPEED)*DSQRT((1.+ALPHA*Z)/(1.+ALPHA*Z+DTOT))
      IF ((DREAL(BETA) .GE. BRAN1) .AND.
     1 (DIMAG(BETA) .LE. 0.0)) THEN
          SQRT1=-CDSQRT(AA*BB-DTOT)
        ELSE
          SQRT1=CDSQRT(AA*BB-DTOT)
        END IF
      RETURN
      END
C
C
C
C
      FUNCTION SQRT2(BETA)
      IMPLICIT DOUBLE PRECISION  (A-H,O-Z)
      COMPLEX*16 BETA,AA,SQRT2
       COMMON /BRANCH/BRO,BRS,BRZ,BRW
       COMMON /CONSTANT1/SPEED,OMEGA
       COMMON /CONSTANT2/ALPHA,DTOT
      AA=1.-(((SPEED/OMEGA)*BETA)*((SPEED/OMEGA)*BETA))
     IF ((DREAL(BETA) .GE. BRW) .AND.
     1 (DIMAG(BETA) .LE..0)) THEN
          SQRT2=-CDSQRT(AA)
        ELSE
          SQRT2=CDSQRT(AA)
        END IF
      RETURN
      END
C
C
C
C
C *******************************************************************
C FUNCTION MODLOG COMPUTES THE LOG OF BETA GIVEN DIRECTION AND      *
C  LOCATION OF BRANCH CUTS.                                         *
C *******************************************************************
      FUNCTION MODLOG(QUAN)
      IMPLICIT DOUBLE PRECISION  (A-H,O-Z)
      COMPLEX*16 QUAN,MODLOG
      IF ((DREAL(QUAN) .LE. 0.0) .AND. (DIMAG(QUAN)
     1  .GE. 0.0)) THEN
        MODLOG=CDLOG(QUAN)+DCMPLX(0.0,-2*3.1415927)
      ELSE
        MODLOG=CDLOG(QUAN)
      END IF
      RETURN
      END
C
C
C  END OF FUNCTIONS USED ABOVE.
C
```

```
C
      SUBROUTINE HALL(Z,H2,H21,H1,H11)
C*****************************************************************
C  HALL USES SUBROUTINE CGBAIR TO CALCULATE 1/3 ORDER            *
C   HANKEL  FUNCTIONS FROM AIRY FUNCTIONS.                       *
C*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMPLEX*16 Z,AI,BI,AIP,BIP,K,KS,H1,H2,H11,H21
      COMPLEX*16 ARG,CI
      CI= DCMPLX(0.D0,1.D0)
      PI= 3.141592654D0
      ARG= DCMPLX(0.D0,-PI/6.D0)
      K= (12.D0)**(1.D0/6.D0)*CDEXP(ARG)
      KS= DCONJG(K)
      CALL CGBAIR(-Z,AI,BI,AIP,BIP)
      H1= K*(AI-CI*BI)
      H2= KS*(AI+CI*BI)
      H11= -K*(AIP-CI*BIP)
      H21= -KS*(AIP+CI*BIP)
      RETURN
      END
C
C
C
C
      SUBROUTINE CGBAIR(Z,AI,BI,AIP,BIP)
C ****************************************************************
C CALCULATE AIRY FUNCTIONS FOR COMPLEX*16 ARGUMENT               *
C  REF. HANDBOOK OF MATHEMATICAL FUNCTIONS, ABRAMOWITZ AND STEGUN. *
C   ENTRY:                                                       *
C   CALCULATE ARGUMENT(Z) AND ABSOLUTE VALUE(Z)                  *
C    IF /Z/ LT 6                                                 *
C    THEN USE EQS. 10.4.2 THRU 10.4.5 FOR AI,BI,AIP,BIP          *
C 10 ELSE IF ARG(Z) LT PI/3                                      *
C     THEN CALCULATE ZETA(Z)                                     *
C     USE EQS. 10.4.59, 10.4.61, 10.4.63, 10.4.66 FOR AI,BI,AIP,BIP*
C 20 ELSE CALCULATE ZETA(-Z)                                     *
C     USE EQS. 10.4.60, 10.4.62, 10.4.64, 10.4.67 FOR AI,BI,AIP,BIP*
C           ENDIF                                                *
C      ENDIF                                                     *
C      EXIT                                                      *
C      END                                                       *
C ****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMPLEX*16 Z,AI,BI,AIP,BIP,ZETA,CZETA,Z14,SUM1,SUM2,SUM3,SUM4,
     1 ZETAP,FACT1,FACT2,SN,CS,FTERM,FPTERM,GTERM,GPTERM,F,FP,G,GP,Z3
       COMPLEX*16 VZETA,VZETAP
      DIMENSION C(21),D(21)
      DATA C1,C2,PIRT,PI4/.3550280539D0,.2588194038D0,1.772453851D0,
     + .7853981635D0/
      DATA C/1.D0,.0694444444444444D0,
     + .0371334876543210D0,.0379930591278000D0,
     1 .057649190412669DO,.11609906402551DO,
     + .29159139923074DO,.87766696950998DO,
     2 3.0794530301731DO,12.341573332345DO,
     + 55.622785365914DO,278.46508077759DO,
     3 1533.1694320127DO,9207.2065997258DO,
     + 59892.513565875DO,419524.87511653DO,
     4 3148257.4178666DO,25198919.871601DO,
     + 214288036.96366DO,1929375549.1823DO,
     5 18335766937.889DO/
       DATA D/1.DO,
     + -.097222222222221DO,-.043885030864197DO,-.042462830789894DO,
     1 -.062662163492031DO,-.12410589602727DO,-.30825376490107DO,
     2 -.92047999241291DO,-3.2104935846485DO,-12.807293080735DO,
     3 -57.508303513911DO,-287.03323710920DO,-1576.3573033370DO,
     4 -9446.3548230953DO,-61335.706663847DO,-428952.40040004DO,
     5 -3214536.5214006DO,-25697908.383909DO,-218293420.83214DO,
     6 -1963523788.9909DO,-18643931088.105DO/
       ABSZ=ABS(Z)
       IF(ABSZ.EQ.0) GO TO 3.
       IF(ABS(DIMAG(Z)).LE.1.D-12.AND.DREAL(Z).LT.0.DO) GO TO 5
       ARGZ=ATAN2(DIMAG(Z),DREAL(Z))
       GO TO 4
    3  ARGZ=0.DO
       GO TO 4
    5  ARGZ=3.141592653589 8DO
    4  CONTINUE
       IF(ABSZ.GT.6.DO) GO TO 10
C      ASCENDING SERIES
C      EQS. 10.4.2,10.4.3
  15    CONTINUE
       Z3=Z**3
       FTERM=1.DO
       FPTERM=Z*Z/2.DO
       GTERM=Z
       GPTERM=1.DO
       GLIM=1.D-13*ABSZ
       F=FTERM
       FP=FPTERM
       G=GTERM
       GP=GPTERM
        KKKT=100 !  ADJUST KKKT TO INSURE CONVERGENCE IF NECESSARY
       DO 1 I=1,KKKT
       I3=3*I
       FTERM=FTERM*Z3/((I3-1.DO)*I3)
       FPTERM=FPTERM*Z3/(I3*(I3+2.DO))
       GTERM=GTERM*Z3/(I3*(I3+1.DO))
       GPTERM=GPTERM*Z3/((I3-2.DO)*I3)
       F=F+FTERM
       FP=FP+FPTERM
       G=G+GTERM
       GP=GP+GPTERM
       IF(ABS(GTERM).LE.GLIM) GO TO 2
    1 CONTINUE
      PRINT 6000, Z
 6000 FORMAT(/' Z='2E14.5,' ERROR IN CGBAIR, NONCONVERGENCE')
    2 AI=C1*F-C2*G
      AIP=C1*FP-C2*GP
      BI=1.732050808D0*(C1*F+C2*G)
      BIP=1.732050808D0*(C1*FP+C2*GP)
      GO TO 9999
C     ASYMPTOTIC EXPANSIONS FOR /Z/ LARGE
   10 SIGN=1.D0
      SUM1=0.D0
      SUM2=0.D0
      SUM3=0.D0
      SUM4=0.D0
       PIBY3=3.1415926D0/3.D0
      IF(ABS(ARGZ).GE.PIBY3) GO TO 20
C     /ARG(Z)/ LE PI/3
C     EQS. 10.4.59, 10.4.61, 10.4.63, 10.4.66
C
      ZETA=CZETA(ABSZ,ARGZ)
      DO 11 I=1,12
      K=I-1
      ZETAP=ZETA**K
      SUM1=SUM1+SIGN*C(I)/ZETAP
      SUM2=SUM2+SIGN*D(I)/ZETAP
      SUM3=SUM3+C(I)/ZETAP
      SUM4=SUM4+D(I)/ZETAP
   11 SIGN=-SIGN
      Z14=ABSZ**.25D0*DCMPLX(COS(ARGZ/4.D0),SIN(ARGZ/4.D0))
      FACT1=.5D0*EXP(-ZETA)/(PIRT*Z14)
      FACT2=.5D0*EXP(-ZETA)*Z14/PIRT
      AI=FACT1*SUM1
      AIP=-FACT2*SUM2
      FACT1=EXP(ZETA)/(PIRT*Z14)
      FACT2=EXP(ZETA)*Z14/PIRT
      BI=FACT1*SUM3
      BIP=FACT2*SUM4
      GO TO 9999
C
C     /ARG(Z)/ GT PI/3 NOTE CHANGE ABOVE
C     EQS. 10.4.60, 10.4.62, 10.4.64, 10.4.67
C
   20  CONTINUE
       ARGZ=ATAN2(-DIMAG(Z),-DREAL(Z))
      ZETA=CZETA(ABSZ,ARGZ)
       VZETA=1.D0/ZETA
       LLL=10
      DO 21 I=1,LLL
      K2=(I-1)*2
      J=K2+1
       VZETAP=VZETA**K2
       SUM1=SUM1+SIGN*C(J)*VZETAP
```

```
        SUM2=SUM2+(SIGN*C(J+1)*VZETAP*VZETA)
        SUM3=SUM3+(SIGN*D(J)*VZETAP)
        SUM4=SUM4+(SIGN*D(J+1)*VZETAP*VZETA)
   21  SIGN=-SIGN
       Z14=ABSZ**.25DO*DCMPLX(COS(ARGZ/4.DO),SIN(ARGZ/4.DO))
       FACT1=1.DO/(PIRT*Z14)
       FACT2=Z14/PIRT
       SN=SIN(ZETA+PI4)
       CS=COS(ZETA+PI4)
       AI=FACT1*(SN*SUM1-CS*SUM2)
       AIP=-FACT2*(CS*SUM3+SN*SUM4)
       BI=FACT1*(CS*SUM1+SN*SUM2)
       BIP=FACT2*(SN*SUM3-CS*SUM4)
 9999  RETURN
       END
C
C
C BEGIN OF FUNCTIONS USED ABOVE
C
C
       FUNCTION CZETA(ABSZ,ARGZ)
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       COMPLEX*16 CZETA
       ARG=ARGZ*1.5DO
       CZETA=(ABSZ**1.5DO)*DCMPLX(COS(ARG),SIN(ARG))*.66666666666667DO
       RETURN
       END
C
C
C  END OF FUNCTIONS USED ABOVE.
C
C
        SUBROUTINE GZALL1(Z,BETA,GZ,GZZ,EN)
C **********************************************************
C  GZALL1 CALCULATES ALL THE PARTIAL DERIVATIVES           *
C   OF THE g FUNCTION.                                     *
C **********************************************************
        IMPLICIT DOUBLE PRECISION(A-H,O-Z)
        COMPLEX *16 BETA,GZ,GZZ,G,GB,GBB,SQRT1,EN
        COMMON /CONSTANT1/SPEED,OMEGA
        COMMON /CONSTANT2/ALPHA,DTOT
        CALL GALL(Z,BETA,G)
        A=1.+ALPHA*Z+DTOT
        BRAN=OMEGA/SPEED*SQRT((A-DTOT)/A)
        IF (DREAL(G).LE..O.AND.DIMAG(G).GE.O.) THEN
                SI=-1.
           ELSE
                SI=1.
        ENDIF
        GZ=SI*2.*ALPHA*SQRT1(BETA,Z)/(3.*CDSQRT(G*A))
        C=2.*ALPHA**3.DO*DTOT/(9*A**2.DO)
        GZZ=C/(GZ*G)-.5*GZ**2.DO/G
       T=(3.*OMEGA/(2.*ALPHA*SPEED))**(2.D0/3.D0)
       EN=T*G
       RETURN
       END
C
C
C
C
       SUBROUTINE GALL(Z,BETA,G)
C *********************************************
C  GALL EVALUATES THE g FUNCTION              *
C *********************************************
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       COMPLEX *16 G,BETA,G32
       COMMON /CONSTANT1/SPEED,OMEGA
       COMMON /CONSTANT2/ALPHA,DTOT
       CALL G32ALL(Z,BETA,G32)
       G=CDEXP(2./3.*CDLOG(G32))
       RETURN
       END
C
C
C
C
      SUBROUTINE HALL2(Z,H2,H1)
C ******************************************************
C  HALL2 USES SUBROUTINE CGBAIR TO CALCULATE 1/3 ORDER *
C   HANKEL  FUNCTIONS FROM AIRY FUNCTIONS. NOT THE     *
C   DERIVATIVES AS HALL DOES.                          *
C*******************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMPLEX*16 Z,AI,BI,AIP,BIP,K,KS,H1,H2,H11,H21
      COMPLEX*16 ARG,CI
       COMPLEX*16 BETA
      CI= DCMPLX(0.D0,1.D0)
      PI= 3.141592654D0
      ARG= DCMPLX(0.D0,-PI/6.D0)
      K= (12.D0)**(1.D0/6.D0)*CDEXP(ARG)
      KS= DCONJG(K)
      CALL CGBAIR(-Z,AI,BI,AIP,BIP)
      H1= K*(AI-CI*BI)
      H2= KS*(AI+CI*BI)
      RETURN
      END
C
C
C
C
       SUBROUTINE DAFB2(F)
C ****************************************************************
C  SUBROUTINE TO ACCURATELY DO THE HANKEL TRANSFORM OF THE SOUND *
C   PRESSURE LEVEL.                                              *
```

```
C   F(NP)=GBAR(NP) MUST BE SAMPLED AT NP POINTS WITH K=(N-1,ALP)       *
C   ALP REPRESENTS THE DISTANCE ABOVE THE REAL AXIS THE FUNCTION WILL *
C   BE INTEGRATED.                                                     *
C   NS IS A PARAMETER REPRESENTING ADDITION OF AN ANALYTICAL FUNCTION *
C   TO F(NP)                                                           *
C   M IS THE NUMBER OF TERMS USED TO APPROXIMATE F(NP) TO INFINITY     *
C***********************************************************************
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        COMMON /INTEG/NS,ME,N1
        COMMON/AFB2IN/ALP,DELBETA
        COMPLEX*16 F(N1),CF,CARG,SUM,FNP,CMPI,D1
        NP=N1
        CMPI=DCMPLX(0.DO,1.DO)
        DELK=DELBETA
        PI=3.1415926DO
C  SUBTRACT THE ANALYTICAL FUNCTION IF NS > ZERO
C   ADJUSTING THE SUBTRACTION MULTIPLIER CF
        IF(NS.LE.0) GOTO 11
        CF=DCMPLX(0.DO,0.DO)
        IF (ALP.EQ.0.0) THEN
         CF=DFLOAT(NP)/DFLOAT(NS)
         CF=CF*F(2)
        END IF
        IF (ALP.NE.0.0) THEN
         CF=CMPI*DFLOAT(NP)*F(1)/(DFLOAT(NS)*ALP)
        END IF
C   SUBTRACT THE ANALYTICAL FUNCTION IF NS>0
        DO 10,I=1,NP
         D1=DCMPLX(DFLOAT(I-1),(-ALP))
         CARG=DFLOAT(NS)*(-D1)/DFLOAT(NP)
        F(I)=F(I)-CF*(1.DO-CDEXP(CARG))
  10    CONTINUE
  11    CONTINUE
        IF(ALP.EQ.0.0) F(1)=DCMPLX(0.DO,0.DO)
        FNP=F(NP)
        DO 12,I=2,NP
         D1=DCMPLX(DFLOAT(I-1),(-ALP))
         F(I)=F(I)/(CDSQRT(D1))
  12    CONTINUE
        IF(ALP.NE.0.0) F(1)=F(1)/CDSQRT((-CMPI)*(ALP))
C ADD TERMS TO INFINITY IF ME>0
        IF(ME.LT.1) GOTO 20
        DO 15,I=1,NP
         D1=DCMPLX(DFLOAT(I-1),(-ALP))
         CF=D1/DFLOAT(NP)
         CALL ZETA(NP,ME,CF,SUM)
         F(I)=F(I)+FNP*SUM
  15    CONTINUE
  20    CONTINUE
C  DO THE FFT
        CALL FORK(NP,F,1)
C ADD ALTERNATE TERMS TO GIVE NP/2 SAMPLES TRANSFORMED
```

```
          CF=DELK*DFLOAT(NP)*(CDSQRT(-CMPI))/(2.DO*PI)
          DO 25,I=2,NP/2
           A=DEXP(DFLOAT(I-1)*(ALP)*2.DO*PI/DFLOAT(NP))
           F(I)=A*F(I)+CMPI*F(NP-I+2)/A
           F(I)=F(I)*CF/DSQRT(DFLOAT(I-1))
   25     CONTINUE
          RETURN
          END
C
C
C
C
          SUBROUTINE FORK(LX,CX,SIGNI)
C ******************************************************************
C  A FAST FFT GIVEN BY J.F. CLAERBOUT, "FUNDAMENTALS OF GEOPHYSICAL *
C   DATA PROCESSING"                                               *
C*******************************************************************
          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
          COMPLEX*16 CX(LX),CARG,CW,CTEMP,CI,DUM1,CMPI
          INTEGER SIGNI
          J=1
          CMPI=DCMPLX(O.DO,1.DO)
          PI=3.1415926
          SC=DSQRT(1.DO/DFLOAT(LX))
          DO 30,I=1,LX
           IF(I.GT.J) GOTO 10
           CTEMP=CX(J)*SC
           CX(J)=CX(I)*SC
           CX(I)=CTEMP
   10              M=LX/2
   20              IF(J.LE.M) GOTO 30
           J=J-M
           M=M/2
           IF(M.GE.1) GOTO 20
   30              J=J+M
          L=1
   40              ISTEP=2*L
          DO 50,M=1,L
           CARG=CMPI*PI*DBLE(SIGNI)*DBLE((M-1))/DBLE(L)
           CW=CDEXP(CARG)
           DO 50,I=M,LX,ISTEP
            CTEMP=CW*CX(I+L)
            CX(I+L)=CX(I)-CTEMP
   50              CX(I)=CX(I)+CTEMP
          L=ISTEP
          IF(L.LT.LX) GOTO 40
          RETURN
          END
C
C
C
C
```

```
        SUBROUTINE ZETA(NP,M,A,SUM)
C********************************************************
C       SUBROUTINE TO ADD THE NECESSARY TERMS TO
C        EXPRESS INVERTIBLE FUNCTION TO INFINITY.
C        WILL USE DOUBLE PRECISION.
C        SUM=SUM OF 1/(NP^.5)*1/((J+A)^.5) FOR J=1 TO
C        INFINITY  MINUS SOME CONSTANT WHICH IS
C        INDEPENDENT OF A.
C********************************************************
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        COMPLEX*16 A,SUM,D2
        D2=DCMPLX(DFLOAT(M),0.DO)
        SUM=1.DO/(M+A)
        SUM=2.DO*(DSQRT(DFLOAT(M))-1.DO/CDSQRT(SUM))
     $   -0.5*CDSQRT(SUM)*(1.0+SUM*(1.0/12.0-SUM*SUM/192.0))
        DO 10,J=1,M
         SUM=SUM+1.DO/CDSQRT(J+A)
  10    CONTINUE
        SUM=SUM/DSQRT(DFLOAT(NP))
        RETURN
        END
```

C-2

APPENDIX B - The Inversion Case Program

The code for the lapse case follows. This code is extremely similar to the lapse case and the subroutine names and functions, variable names and hints are identical or at least very similar to those in the lapse case.

ORIGINAL PAGE IS
OF POOR QUALITY

```
      PROGRAM MAIN
C
C
C     ******************************************************************
C     *                                                                *
C     *       Main program: will call the subroutines to:              *
C     *        1)  Input the program parameters.                       *
C     *        2)  Find the points in the g^3/2 function where         *
C     *            region changes occur.                               *
C     *        3)  Build the matrix of values obtained by marching     *
C     *            slightly above the real axis.                       *
C     *        4)  Perform the Hankel Transform on the matrix          *
C     *            obtained in step three.                             *
C     *        5)  Print the results.                                  *
C     *                                                                *
C     ******************************************************************
C
C
C
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /MAIN/    F
      COMMON /CONST/   CI,PI
      COMMON /CONST1/  SPEED,OMEGA
      COMMON /CONST2/  ALPHA,DTOT
      COMMON /CONST3/  Z,ZREF,S
      COMMON /REGION/  RS1,RS2,RZ1,RZ2,RO1,RO2
      COMMON /STATE/   RCE,RESISTANCE
      COMMON /INTEG/   NS,ME,NOPTS
      COMMON /AFEGIN/  HGHT,DELBETA
      COMPLEX*16 F(32768),CI,GUESS,ROOT,G32
      CI     = DCMPLX(0.D0,1.D0)
      PI     = DATAN2(0.D0,-1.D0)
      PRINT *,'PI= ',PI
C
C
      CALL INPUT(DELBETA,HGHT,RT1,RT2)
      ANG    = -PI/2.D0
C
C
        PRINT *,'DETERMINING REGION CHANGE COORDINATES'
      CALL REGION_FIND(ANG,RT1,RT2)
C
C
C
        PRINT *,'BUILDING MATRIX'
      CALL BUILDMATRIX (DELBETA,HGHT)
C
C
C
        PRINT *,'DOING HANKEL TRANSFORM'
      CALL HANKEL
C
C
C
      print *,'Printing results to FOR045.DAT'
      CALL PRINTALL(DELBETA,NOPTS)
C
C
      PRINT *,'** COMPLETE **'
      STOP
      END
C
C
C
C
      SUBROUTINE INPUT(DELBETA,HGHT,RT1,RT2)
C
C     ***********************************************************
C     *                                                         *
C     *       THIS IS THE INPUT SECTION OF THE PROGRAM          *
C     *       ALL PARAMETERS NECESSARY FOR THE OPERATION        *
C     *       OF THE PROGRAM WILL BE INPUT IN THIS ROUTINE      *
C     *                                                         *
C     ***********************************************************
C
C
C
C
      IMPLICIT DOUBLE FRECISION (A-H,O-Z)
      COMMON /CONST/  CI,PI
      COMMON /CONST1/ SPEEC,OMEGA
      COMMON /CONST2/ ALPHA,DTCT
      COMMON /CONST3/ Z,ZREF,S
      COMMON /STATE/  ROE,RESISTANCE
      COMMON /INTEG/  NS,ME,NOPTS
      COMPLEX*16 CI
C
      WRITE (6,80C)
      PRINT *,'INPUT PHYSICAL PARAMETERS: Cetector height,Source height,'
      PRINT *,'                            and Angular Frequency'
      READ *,Z,S,OMEGA
      PRINT *,'INPUT STATE PARAMETERS:Temp gradient,Temperature profile'
      print *,'                         constant and Temp. at infinity'
      READ *,DTOT,ALPHA,TINF
      PRINT *,'Input the chessell model FLOW RESISTANCE'
      READ *,RESISTANCE
 1    PRINT *,'Input the  NUMBER OF DATA PCINTS'
      READ *,NCPTS
      TST =CLOG(DFLOAT(NCPTS))/CLOG(2.CO)
      IF(DABS(TST-CNINT(TST)) .GT. 1.0-8) THEN
        NOPTS=DNINT(2.CO**CNINT(TST))
        PRINT *,'Invalid input  (must be a power of 2) TRY AGAIN'
        PRINT *,'               (ex. ',NOPTS,')'
        GOTO 1
      END IF
      PRINT *,'INPUT HANKEL TRANSFORM PARAMETERS:  NS, ME and Integration
     & Height'
      READ *,NS,ME,HGHT
      ZREF    = 0.CO
      SPEED   = DSCRT(401.8D0 * TINF)
      RT1     = OMEGA/SPEEC
      RT2     = RT1*DSCRT(1.D0/(1.CO+DTOT))
      CELBETA = RT2 * 1.01D0 /NCPTS
      WRITE (6,800)
 80C  FORMAT('0',2CX,1H,///////////////////////////////)    ! clears the screen
      PRINT *,'Reference hght =',ZREF
      PRINT *,'Detector height=',Z
      PRINT *,'Source height  =',S
      PRINT *,'Frequency (rad)=',OMEGA
      PRINT *,'speed of sound =',SPEED
      PRINT *,'Temp Gradient  =',DTOT
      PRINT *,'Profile (alpha)=',ALPHA
      PRINT *,'T infinity     =',Tinf
      PRINT *,'Chessel resist.=',RESISTANCE
      PRINT *,'Num. points    =',NOPTS
      PRINT *,'Del Beta       =',DELBETA
      PRINT *,'----------'
      PRINT *,' Hankel'
      PRINT *,'constants   NS  =',NS
      PRINT *,'            ME  =',ME
      print *,'           Alp  =',HGHT
      PRINT *,'integration hght=',HGHT*DELBETA
      PRINT *,'.'
      RETURN
      END
C
C
C
C
C
      SUBROUTINE BUILDMATRIX(DELBETA,ALP)
C
C     ******************************************************************
C     *                                                                *
C     *       This subroutine is used to determine the function        *
c     *       "G" in Van Moornem's notes. It will be called by         *
c     *       the main program and will return the matrix which        *
c     *       will be inverted by the Hankel program.                  *
c     *                                                                *
c     ******************************************************************
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON /MAIN/   GBAR
      COMMON /CONST/  CI,PI
      COMMON /CONST1/ SPEED,OMEGA
      COMMON /CONST2/ ALPHA,DTOT
      COMMON /CONST3/ Z,ZREF,S
      COMMON /REGION/ RS1,RS2,RZ1,RZ2,RO1,RO2
      COMMON /STATE/  ROE,RESISTANCE
      COMMON /INTEG/  NS,ME,NOPTS
      COMPLEX*16 GBAR(32768),BETA
      COMPLEX*16 GZ_Z,GZ_S,GZ_C,GZZ_O,GZZ,GZ
      COMPLEX*16 ETA_Z,ETA_S,ETA_O
      COMPLEX*16 H1_NZ,H2_NZ,H1_NS,H2_NS
      COMPLEX*16 H1_NO, H2_NO, H11_NO, H21_NO
      COMPLEX*16 CONST,R,E,CI,T1,T2,T3
      COMPLEX*16 TAU,PSI,ZIMP
      COMPLEX*16 SRGZZ,SRGZS,SRGZ,SRGS
      INTEGER IREGION
      E       = CDEXP(- PI * CI / 3.D0)
      FR      = OMEGA/(2.D0*PI)
      RATIO   = FR/RESISTANCE
      DO      = 1.D0+9.08D0*RATIO**(-.75D0)
      X       = -11.9D0*RATIO**(-.73D0)
      ZIMP    = DCMPLX(DO,X)
        Q     = 1.D0
C       Q     = 0.00002*4.D0*PI*4.67D0*(10.D0**(SPLREF/20.D0))
      RLMDA   = OMEGA/(SPEED*ALPHA)
      RLMDA23 = RLMDA ** (2.D0/3.D0)
C
C
```

```
C
C
C                     ******************************
C                     **    PROGRAM BEGINS HERE    **
C                     ******************************
        DO 10,I=1,NOPTS
C
        BETA     = DCMPLX (DFLOAT(I-1),(ALP))*(DELBETA)
        CALL GZALL1(ZREF,BETA,GZ_C,GZZ_O,ETA_O)
        CALL GZALL1(Z,BETA,GZ_Z,GZZ,ETA_Z)
        CALL GZALL1(S,BETA,GZ_S,GZZ,ETA_S)
        CALL HALL(ETA_O,H2_NO,H21_NO,H1_NO,H11_NO,IREGO)
        CALL HALL2(ETA_S,H2_NS,H1_NS,IREGS)
        CALL HALL2(ETA_Z,H2_NZ,H1_NZ,IREGZ)
        CALL REGION (BETA,IREGION)
          TAU    = 1.D0 - (SPEED/OMEGA * CI/2 * ZIMP * GZZ_O/GZ_O)
          PSI    = ZIMP*CI*SPEED/OMEGA
          PSI    = PSI * ((3.D0/2.D0 * RLMDA)**(2.D0/3.D0)) * GZ_C
          SRGZ   = CDSQRT(GZ_Z)
          SRGS   = CDSQRT(GZ_S)
          CONST = Q * BETA / (12*CI*(RLMDA23)*SRGZ*SRGS)
          IF ((ABS(H2_NO)) .EQ. 0.D0 .AND. (ABS(H21_NO) .EQ. 0.D0)) THEN
               R=DCMPLX(1.D35,0.D0)
                GOTO 22
          END IF
          R      =    TAU * H2_NO + PSI * H21_NO
          R      = -(TAU * H1_NO + PSI * H11_NO) / R
  22    GO TO(1,2,3,4,5,6,7,8),IREGION
C
C
C
!                  *    REGION 1   *
   1    GBAR(I)  = CONST * (H1_NS + H2_NS*R) * H2_NZ
        GOTO 10
!
!                  *    REGION 2   *
   2    GBAR(I)  = CONST * (H1_NZ + H2_NZ*R) * H2_NS
        GOTO 10
!
!                  *    REGION 3   *
   3    GBAR(I)  =-(CONST /(E*E)) * (H1_NS/R + H2_NS) * H2_NZ
        GOTO 10
!
!                  *    REGION 4   *
   4    GBAR(I)  =-CONST * (H1_NS/R + H2_NS) * H1_NZ
        GOTO 10
!
!                  *    REGION 5   *
   5    GBAR(I)  =-CONST * (H1_NZ/R + H2_NZ) * H1_NS
        GOTO 10
!
!                  **    REGION 6   **
   6    GBAR(I)  = CONST * (H1_NS + ((R-E)/R)*E * H2_NS) * H2_NZ
        GOTO 10
!
!                  *    REGION 7   *
   7    GBAR(I)  = CONST * (H1_NZ + ((R-E)/R)*E * H2_NZ) * H2_NS
        GOTO 10
!
!                  *    REGION 8   *
```

```
  8     GBAR(I)  = CONST * E * (H1_NZ/R + H2_NZ) * H2_NS
C
C
 10     CONTINUE
        RETURN
        END
C
C
C
C
        SUBROUTINE G32ALL(Z,BETA,G32C)
C*******************************************************************
C*                                                                 *
C*               G32ALL CALCULATES g3/2 FUNCTION                   *
C*                                                                 *
C*                                                                 *
C*      modified so the branch point of the LOG function           *
C*      is below the positive real axis. Feb 20, 86                *
C*         modified again 10/12/86 for modified program            *
C*******************************************************************
        IMPLICIT DOUBLE PRECISION  (A-H,O-Z)
        COMPLEX*16 BETA,PHI,SQRT1,SQRT2,FOO,G32C,MODLOG
        COMPLEX*16 S1,A,B,S,CI
        COMMON /CONST/  CI,PI
        COMMON /CONST1/ SPEED,OMEGA
        COMMON /CONST2/ ALPHA,DTOT
        AA   = 1.D0 + ALPHA*Z + DTOT
        B    = 1.D0 - ((SPEED*BETA/OMEGA) * (SPEED*BETA/OMEGA))
        S1   = SQRT1(BETA,Z)
        S    = SQRT2(BETA)
        PHI = S1 /(S*DSQRT(AA))
        IF(DTOT .EQ. .0 .OR. CDABS(1.-PHI) .LT.  1.D-8) THEN
                FOO= 0.0
        ELSE
                FOO= .5 * CDLOG( (1.+PHI)/(1.-PHI) )
        ENDIF
        G32C  = - DSQRT(AA) + S1  +  DTOT * FOO/S
        RETURN
        END
C
C
C
C BEGIN OF FUNCTIONS USED ABOVE.
C
C ********************************************************************
C SQRT1 AND SQRT2 ARE FUNCTIONS TO CALCULATE SQRT(BETA) GIVEN        *
C  DESIRED BRANCH CUTS AND DIRECTION +IMAGINARY OR -IMAGINARY.       *
C  *******************************************************************
        FUNCTION SQRT1(BETA,Z)
        IMPLICIT DOUBLE PRECISION  (A-H,O-Z)
        COMPLEX*16 BETA,AA,SQRT1,CI
        COMMON /CONST/ CI,PI
        COMMON /CONST1/SPEED,OMEGA
        COMMON /CONST2/ALPHA,DTOT
        AA    = 1.-(((SPEED/OMEGA)*BETA)*((SPEED/OMEGA)*BETA))
        BB    = 1. + ALPHA*Z + DTOT
        SQRT1=CDSQRT (AA*BB-DTOT)
        RETURN
        END
C
```

```
C
C
C
      FUNCTION SQRT2(BETA)
      IMPLICIT DOUBLE PRECISION  (A-H,O-Z)
      COMPLEX*16 BETA,AA,SQRT2,CI
       COMMCN /CONST/  CI,PI
       COMMCN /CONST1/ SPEEC,OMEGA
       COMMCN /CONST2/ ALPHA,DTCT
         AA     = 1.-(((SPEED/OMEGA)*BETA)*((SPEED/OMEGA)*BETA))
         SQRT2 =  CDSQRT(AA)
      RETURN
      END
C
C
C
C
       SUBROUTINE GALL(Z,BETA,G)
C **************************************************
C  GALL EVALUATES THE g FUNCTION                   *
C **************************************************
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       COMPLEX *16 G,BETA,G32,B2,CI,g1
       COMMCN /CONST/  CI,PI
       COMMCN /CONST1/ SPEED,OMEGA
       COMMCN /CONST2/ ALPHA,DTOT
       CALL G32ALL(Z,BETA,G32)
       PHI  = DATAN2(DIMAG(G32),DREAL(G32))
       if (phi .gt. 0.dC) phi=phi-2*pi
       G     = ((cdabs(g32))**(2.d0/3.d0)) * cdexp((phi*2.d0/3.dC)*ci)
       g1    = g
       IF (PHI .LE. -PI/2.DC)     THEN
            G = G * CDEXP(2.DC/3.DO*PI*CI)
          ELSE
            G = G * CDEXP(4.DC/3.DO*PI*CI)
          END IF
       RETURN
       END
C
C
C
C
C
       SUBRCUTINE GZALL1(Z,BETA,GZ,GZZ,EN)
C *********************************************************
C  GZALL1 CALCULATES ALL THE PARTIAL DERIVATIVES          *
C   CF THE g FUNCTION.                                    *
C *********************************************************
       IMPLICIT DOUBLE FRECISION(A-H,O-Z)
       COMPLEX *16 BETA,GZ,GZZ,G,GB,GBB,SQRT1,EN,CI,DUM
       COMMON /CONST/ CI,PI
       COMMCN /CONST1/SFEED,CMEGA
       COMMON /CONST2/ALPHA,CTOT
       CALL GALL(Z,BETA,G)
       SI    = 1.DO
       A     = 1.+ALPHA*Z+DTOT
       GZ    = 2.DO * SI* ALPHA * SQRT1(BETA,Z) / (3.*CDSQRT(G*A))
       C     = 2.*ALPHA**3.DC*DTOT/(9*A**2.DO)
       GZZ   = C/(GZ*G)-.5*GZ**2.DO/G
       T     = (3.*OMEGA/(2.*ALPHA*SPEED))**(2.DO/3.DO)
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
          EN    = T*G
          RETURN
          END
C
C
C
C
        SUBROUTINE HALL(Z,H2,H21,H1,H11,IREG)
C*********************************************************************
C  HALL USES SUBROUTINE CGBAIR TO CALCULATE 1/3 ORDER                *
C   HANKEL  FUNCTIONS FROM AIRY FUNCTIONS.                           *
C*********************************************************************
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        COMPLEX*16 Z,AI,BI,AIP,BIP,K,KS,H1,H2,H11,H21
        COMPLEX*16 ARG,CI
        COMMON /CONST/ CI,PI
        ARG = DCMPLX(0.D0,-PI/6.D0)
        K   = (12.D0)**(1.D0/6.D0)*CDEXP(ARG)
        KS  = DCONJG(K)
        CALL CGBAIR(-Z,AI,BI,AIP,BIP,IREG)
        H1  = K*(AI-CI*BI)
        H2  = KS*(AI+CI*BI)
        H11 = -K*(AIP-CI*BIP)
        H21 = -KS*(AIP+CI*BIP)
        RETURN
        END
C
C
C
C
        SUBROUTINE CGBAIR(Z,AI,BI,AIP,BIP,IREG)
C ********************************************************************
C CALCULATE AIRY FUNCTIONS FOR COMPLEX*16 ARGUMENT                   *
C  REF. HANDBOOK OF MATHEMATICAL FUNCTIONS, ABRAMOWITZ AND STEGUN.   *
C   ENTRY:                                                           *
C   CALCULATE ARGUMENT(Z) AND ABSOLUTE VALUE(Z)                      *
C    IF /Z/ LT 6                                                     *
C    THEN USE EQS. 10.4.2 THRU 10.4.5 FOR AI,BI,AIP,BIP              *
C 10 ELSE IF ARG(Z) LT PI/3                                          *
C      THEN CALCULATE ZETA(Z)                                        *
C      USE EQS. 10.4.59, 10.4.61, 10.4.63, 10.4.66 FOR AI,BI,AIP,BIP*
C 20 ELSE CALCULATE ZETA(-Z)                                         *
C      USE EQS. 10.4.60, 10.4.62, 10.4.64, 10.4.67 FOR AI,BI,AIP,BIP*
C             ENDIF                                                  *
C       ENDIF                                                        *
C       EXIT                                                         *
C       END                                                          *
C ********************************************************************
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        COMMON /CONST/ CI,PI
        COMPLEX*16 Z,AI,BI,AIP,BIP,ZETA,CZETA,Z14,SUM1,SUM2,SUM3,SUM4,
      1 ZETAP,FACT1,FACT2,SN,CS,FTERM,FPTERM,GTERM,GPTERM,F,FP,G,GP,Z3
         COMPLEX*16 VZETA,VZETAP,CI
        DIMENSION C(21),D(21)
        DATA C1,C2,PIRT,PI4/.3550280539D0,.2588194038D0,1.772453851D0,
      + .7853981635D0/
         PIRT=DSQRT(PI)
        DATA C/1.D0,.0694444444444444D0,
      + .0371334876543210D0,.0379930591278C0D0,
      1 .0576491904126690D0,.1160990640255100D0,
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
     + .29159139923074D0,.87766696950998D0,
     2 3.0794530301731D0,12.341573332345D0,
     + 55.622785365914D0,278.46508077759D0,
     3 1533.1694320127D0,9207.2065997258D0,
     + 59892.513565875D0,419524.87511653D0,
     4 3148257.4178666D0,25198919.871601D0,
     + 214288036.96366D0,1929375549.1823D0,
     5 18335766937.889D0/
       DATA D/1.D0,
     + -.09722222222222210D0,-.04388503086419700D0,-.04246283078989400D0,
     1 -.06266216349203100D0,-.12410589602727D0,-.30825376490107D0,
     2 -.92047999241291D0,-3.2104935846485D0,-12.807293080735D0,
     3 -57.508303513911D0,-287.03323710920D0,-1576.3573033370D0,
     4 -9446.3548230953D0,-61335.706663847D0,-428952.40040004D0,
     5 -3214536.5214006D0,-25697908.383909D0,-218293420.83214D0,
     6 -1963523788.9909D0,-18643931088.105D0/
        ABSZ = ABS(Z)
       IF(ABSZ.EQ.0) GO TO 3
       IF(ABS(DIMAG(Z)).LE.1.D-12.AND.DREAL(Z).LT.0.D0) GO TO 5
       ARGZ = DATAN2(DIMAG(Z),DREAL(Z))
       GO TO 4
     3 ARGZ = 0.D0
       GO TO 4
     5 ARGZ = PI
     4 CONTINUE
       IF(ABSZ.GT.6.D0) GO TO 10
        ireg=1
C      ASCENDING SERIES
C      EQS. 10.4.2,10.4.3
  15    CONTINUE
       Z3     = Z**3
       FTERM  = 1.D0
       FPTERM= Z*Z/2.D0
       GTERM  = Z
       GPTERM= 1.D0
       GLIM   = 1.D-13*ABSZ
       F      = FTERM
       FP     = FPTERM
       G      = GTERM
       GP     = GPTERM
        KKKT = 100   !  ADJUST KKKT TO INSURE CONVERGENCE IF NECESSARY
       DO 1 I=1,KKKT
       I3     = 3*I
       FTERM = FTERM*Z3/((I3-1.D0)*I3)
       FPTERM=FPTERM*Z3/(I3*(I3+2.D0))
       GTERM = GTERM*Z3/(I3*(I3+1.D0))
       GPTERM=GPTERM*Z3/((I3-2.D0)*I3)
       F      = F+FTERM
       FP     = FP+FPTERM
       G      = G+GTERM
       GP     = GP+GPTERM
       IF(CDABS(GTERM).LE.GLIM) GO TO 2
     1 CONTINUE
       PRINT 6000, Z
  6000 FORMAT(/' Z='2E14.5,' ERROR IN CGBAIR, NONCONVERGENCE')
     2 AI     = C1*F-C2*G
       AIP    = C1*FP-C2*GP
       BI     = 1.7320508D0*(C1*F+C2*G)
       BIP    = 1.7320508D0*(C1*FP+C2*GP)
       GO TO 9999
```

```
C       ASYMPTOTIC EXPANSIONS FOR /Z/ LARGE
   10   SIGN  = 1.D0
        SUM1  = 0.D0
        SUM2  = 0.D0
        SUM3  = 0.D0
        SUM4  = 0.D0
        IF(ABS(ARGZ) .GE. PI/3.D0) GO TO 20
         ireg=2
C       /ARG(Z)/ LE PI/3
C       EQS. 10.4.59, 10.4.61, 10.4.63, 10.4.66
C
        ZETA    = CZETA(ABSZ,ARGZ)
        DO 11 I=1,12
        K       = I-1
        ZETAP   = ZETA**K
        SUM1    = SUM1+SIGN*C(I)/ZETAP
        SUM2    = SUM2+SIGN*D(I)/ZETAP
        SUM3    = SUM3+C(I)/ZETAP
        SUM4    = SUM4+D(I)/ZETAP
        SIGN    = -SIGN
   11   Continue
        Z14     = ABSZ **.25D0 * DCMPLX(COS(ARGZ/4.D0),SIN(ARGZ/4.D0))
        FACT1   = .5D0* CDEXP(-ZETA)/(PIRT*Z14)
        FACT2   = .5D0* CDEXP(-ZETA)*Z14/PIRT
        AI      = FACT1*SUM1
        AIP     = -FACT2*SUM2
        FACT1   = CDEXP(ZETA)/(PIRT*Z14)
        FACT2   = CDEXP(ZETA)*Z14/PIRT
        BI      = FACT1*SUM3
        BIP     = FACT2*SUM4
        GO TO 9999
C
C       /ARG(Z)/ GT PI/3 NOTE CHANGE ABOVE
C       EQS. 10.4.60, 10.4.62, 10.4.64, 10.4.67
C
   20    CONTINUE
         ireg=3
         ARGZ    = DATAN2(-DIMAG(Z),-DREAL(Z))
         ZETA    = CZETA(ABSZ,ARGZ)
         VZETA   = 1.D0/ZETA
         LLL     = 10
         DO 21 I=1,LLL
         K2      = (I-1)*2
         J       = K2+1
         VZETAP  = VZETA**K2
         SUM1    = SUM1+SIGN*C(J)*VZETAP
         SUM2    = SUM2+(SIGN*C(J+1)*VZETAP*VZETA)
         SUM3    = SUM3+(SIGN*D(J)*VZETAP)
         SUM4    = SUM4+(SIGN*C(J+1)*VZETAP*VZETA)
        SIGN     = -SIGN
   21   Continue
        Z14      = ABSZ**.25D0*DCMPLX(COS(ARGZ/4.D0),SIN(ARGZ/4.D0))
        FACT1    = 1.D0/(PIRT*Z14)
        FACT2    = Z14/PIRT
        SN       = SIN(ZETA+PI/4.D0)
        CS       = COS(ZETA+PI/4.D0)
        AI       = FACT1*(SN*SUM1-CS*SUM2)
        AIP      = -FACT2*(CS*SUM3+SN*SUM4)
        BI       = FACT1*(CS*SUM1+SN*SUM2)
        BIP      = FACT2*(SN*SUM3-CS*SUM4)
```

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

```
C
 9999   continue
         RETURN
        END
C
C
C BEGIN OF FUNCTIONS USED ABOVE
C
C
        FUNCTION CZETA(ABSZ,ARGZ)
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        COMPLEX*16 CZETA
        ARG    = ARGZ*1.5D0
        CZETA = (ABSZ**1.5D0)*DCMPLX(COS(ARG),SIN(ARG))*.6666666666666700
        RETURN
        END
C
C
C  END OF FUNCTIONS USED ABOVE.
C
C
C
C
C
        SUBROUTINE HALL2(Z,H2,H1,IREG)
C **********************************************************
C  HALL2 USES SUBROUTINE CGBAIR TO CALCULATE 1/3 ORDER       *
C   HANKEL  FUNCTIONS FROM AIRY FUNCTIONS. NOT THE           *
C   DERIVATIVES AS HALL DOES.                                *
C***********************************************************
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
         COMMON /CONST/ CI,PI
        COMPLEX*16 Z,AI,BI,AIP,BIP,K,KS,H1,H2,H11,H21
        COMPLEX*16 ARG,CI
         COMPLEX*16 BETA
        ARG    = DCMPLX(0.D0,-PI/6.D0)
        K      = (12.D0)**(1.D0/6.D0)*CDEXP(ARG)
        KS     = DCONJG(K)
        CALL CGBAIR(-Z,AI,BI,AIP,BIP,IREG)
        H1     = K*(AI-CI*BI)
        H2     = KS*(AI+CI*BI)
        RETURN
        END
C
C
C
C
         SUBROUTINE HANKEL
C *******************************************************************
C  SUBROUTINE TO ACCURATELY DO THE HANKEL TRANSFORM OF THE SOUND      *
C   PRESSURE LEVEL.                                                   *
C   F(NP)=GBAR(NP) MUST BE SAMPLED AT NP POINTS WITH K=(N-1,ALP)      *
C   ALP REPRESENTS THE DISTANCE ABOVE THE REAL AXIS THE FUNCTION WILL *
C   BE INTEGRATED.                                                    *
C   NS IS A PARAMETER REPRESENTING ADDITION OF AN ANALYTICAL FUNCTION *
C   TO F(NP)                                                          *
C   M IS THE NUMBER OF TERMS USED TO APPROXIMATE F(NP) TO INFINITY    *
C********************************************************************
         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
         COMMON /CONST/ CMPI,PI
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      COMMON /MAIN/ F
      COMMON /INTEG/NS,ME,N1
      COMMON/AFB2IN/ALF,DELBETA
      COMPLEX*16 F(32768),CF,CARG,SUM,FNP,CMPI,D1
      ALP   = -ALP     !   necessary to watch attenborough's definition of alp
      NP    = N1
      DELK = DELBETA
C  SUBTRACT THE ANALYTICAL FUNCTION IF NS > ZERO
C   ADJUSTING THE SUBTRACTION MULTIPLIER CF
      IF(NS.LE.0) GOTO 11
      CF    = DCMPLX(0.D0,0.D0)
      IF (ALP.EQ.0.0) THEN
       CF   = DFLOAT(NP)/DFLOAT(NS)
       CF   = CF*F(2)
      END IF
      IF (ALP.NE.0.0) THEN
       CF   = CMPI*DFLOAT(NP)*F(1)/(DFLOAT(NS)*ALP)
      END IF
C   SUBTRACT THE ANALYTICAL FUNCTION IF NS>0
      DO 10,I=1,NP
       D1   = DCMPLX(DFLOAT(I-1),(-ALP))
       CARG= DFLOAT(NS)*(-D1)/DFLOAT(NP)
      F(I) = F(I)-CF*(1.D0-CDEXP(CARG))
 10   CONTINUE
 11   CONTINUE
      IF(ALP.EQ.0.0) F(1)=DCMPLX(0.D0,0.D0)
      FNP   = F(NP)
      DO 12,I=2,NP
       D1   = DCMPLX(DFLOAT(I-1),(-ALP))
       F(I)= F(I)/(CDSQRT(D1))
 12   CONTINUE
      IF(ALP.NE.0.0) F(1)=F(1)/CDSQRT((-CMPI)*(ALP))
C ADD TERMS TO INFINITY IF ME>0
      IF(ME.LT.1) GOTO 20
      DO 15,I=1,NP
       D1   = DCMPLX(DFLOAT(I-1),(-ALP))
       CF   = D1/DFLOAT(NP)
       CALL ZETA(NP,ME,CF,SUM)
       F(I)= F(I)+FNP*SUM
 15   CONTINUE
 20   CONTINUE
C  DO THE FFT
      CALL FORK(NP,F,1)
C ADD ALTERNATE TERMS TO GIVE NP/2 SAMPLES TRANSFORMED
      CF    = DELK*DFLOAT(NP)*(CDSQRT(-CMPI))/(2.D0*PI)
      DO 25,I=2,NP/2
       A    = DEXP(DFLOAT(I-1)*(ALP)*2.D0*PI/DFLOAT(NP))
       F(I)= A*F(I)+CMPI*F(NP-I+2)/A
       F(I)= F(I)*CF/DSQRT(DFLOAT(I-1))
 25   CONTINUE
      RETURN
      END
C
C
C
C
      SUBROUTINE FORK(LX,CX,SIGNI)
C ******************************************************************
C  A FAST FFT GIVEN BY J.F. CLAERBOUT, "FUNDAMENTALS OF GEOPHYSICAL *
C   DATA PROCESSING"                                                *
```

```
C**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /CONST/ CMPI,PI
      COMPLEX*16 CX(LX),CARG,CW,CTEMP,CI,DUM1,CMPI
      INTEGER SIGNI
      J      = 1
      SC     = DSQRT(1.D0/DFLOAT(LX))
      DO 30,I=1,LX
       IF(I.GT.J) GOTO 10
       CTEMP = CX(J)*SC
       CX(J) = CX(I)*SC
       CX(I) = CTEMP
 10             M=LX/2
 20             IF(J.LE.M) GOTO 30
       J      = J-M
       M      = M/2
       IF(M.GE.1) GOTO 20
 30      J    = J+M
         L    = 1
 40             ISTEP=2*L
      DO 50,M=1,L
       CARG=CMPI*PI*DBLE(SIGNI)*DBLE((M-1))/DBLE(L)
       CW=CDEXP(CARG)
       DO 50,I=M,LX,ISTEP
        CTEMP= CW*CX(I+L)
        CX(I+L)=CX(I)-CTEMP
 50             CX(I)=CX(I)+CTEMP
      L=ISTEP
      IF(L.LT.LX) GOTO 40
      RETURN
      END
C
C
C
C
      SUBROUTINE ZETA(NP,M,A,SUM)
C*************************************************************
C     SUBROUTINE TO ADD THE NECESSARY TERMS TO
C      EXPRESS INVERTIBLE FUNCTION TO INFINITY.
C      WILL USE DOUBLE PRECISION.
C      SUM=SUM OF 1/(NP^.5)*1/((J+A)^.5) FOR J=1 TO
C      INFINITY  MINUS SOME CONSTANT WHICH IS
C      INDEPENDENT OF A.
C*************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMPLEX*16 A,SUM,D2
      D2    = DCMPLX(DFLOAT(M),0.D0)
      SUM   = 1.D0/(M+A)
      SUM   = 2.D0*(DSQRT(DFLOAT(M))-1.D0/CDSQRT(SUM))
     $  -0.5*CDSQRT(SUM)*(1.0+SUM*(1.0/12.0-SUM*SUM/192.0))
      DO 10,J=1,M
       SUM = SUM+1.D0/CDSQRT(J+A)
 10   CONTINUE
      SUM   = SUM/DSQRT(DFLOAT(NP))
      RETURN
      END
C
C
C
C
```

```
      SUBROUTINE PRINTALL(DELBETA,N)
C
C     *********************************************************
C     *                                                       *
C     *       SUBROUTINE TO PRINT THE RESULTS OF THE          *
C     *       HANKEL TRANSFORM TO THE FILE FOR044.DAT         *
C     *                                                       *
C     *********************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /CONST/ CI,PI
      COMMON /MAIN/ F
      COMPLEX*16 F(32768),CI
      DO 100,I=1,N/2
      RAD      = 2.D0*PI* DFLOAT(I-1) / ( DFLOAT(N)*DELBETA )
      IF (RAD .GT. 0.0) THEN
        DECIBLE  = 20.D0 * DLOG10(CDABS(F(I)))
        RAD2     = DLOG10(RAD)
        WRITE (45,900) RAD,RAD2,DECIBLE
      END IF
  100 CONTINUE
  900 FORMAT (5X,3G15.7)
      RETURN
      END
C
C
C
C
      SUBROUTINE REGION_FIND (ANG,RT1,RT2)
C
C     *****************************************************************
C     *                                                               *
c     *       This subroutine determines where region changes         *
c     *       in the g3/2 function take place. The routine will       *
c     *       be called by the main program and will return           *
c     *       the variables RS1,RS2,RZ1,RZ2,RO1 and RO2 which         *
c     *       are the values of beta where region changes occur       *
c     *                                                               *
c     *****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /CONST/  CI,PI
      COMMON /CONST1/ SPEED,OMEGA
      COMMON /CONST2/ ALPHA,DTCT
      COMMON /CONST3/ Z,ZREF,S
      COMMON /REGION/ RS1,RS2,RZ1,RZ2,RO1,RO2
      COMMON /AFB2IN/ HGHT,DELBETA
      COMMON /INTEG/  NS,ME,NOPTS
      COMPLEX*16 B,CI
      DIMENSION RS(3),RZ(3),RO(3)
      print *,'first root is',RT1
      print *,'second root  ',RT2
      HLIM  = RT2 * 1.D100
      BLIM  = REAL(INT(.95D0*RT1/DELBETA)) * DELBETA
    5 H     = HGHT*DELBETA
      I     = 0
      J     = 0
      K     = 0
      B     = DCMPLX(BLIM,H)
   10 CALL G32REGION(S,B,I,ANG,RS1)
```

```
      CALL G32REGION(Z,B,J,ANG,PZ1)
      CALL G32REGION(O,B,K,ANG,PO1)
      RS(I) = REAL(B)
      RZ(J) = REAL(B)
      RO(K) = REAL(B)
      B     = B + DELBETA
      IF ((DREAL(B) .LE. HLIM) .AND. (I+J+K .LT. 9)) GOTO 10
           print *,I,J,K
      IF (I+J+K .LT. 9)PRINT *,'          REGION BOUNDARY NOT DETECTED,'
 55   RS1=RS(1)
      RS2=RS(2)
      RZ1=RZ(1)
      RZ2=RZ(2)
      RO1=RO(1)
      RO2=RO(2)
   6  NT = DINT((MIN(RS2,RZ2,RO2)-RO1)/DELBETA)
      if (nt .gt. 0) then
      PRINT *,'est. Minimum number of points in region  2  is',NT
      end if
   1  print *,'RS1,RS2  ',RS1,RS2
      print *,'RZ1,RZ2  ',RZ1,RZ2
      print *,'RO1,RO2  ',RO1,RO2
      RETURN
      END
C
C
C
      SUBROUTINE G32REGION(Z,B,L,ANG,P1)
C     ************************************************************************
C     * SUBROUTINE TO DETERMINE G32 FUNCTION AND WHERE REGION CHANGES OCCUR *
C     ************************************************************************
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON /CONST/ CI,PI
      COMPLEX*16 B,G32,CI
      CALL G32ALL(Z,B,G32)
      P2    = DATAN2(DIMAG(G32),DREAL(G32))
      IF (P2 .GT. 0.D0) P2=P2-2*PI
          IF (L .EQ. 0) THEN
           L=1
           GOTO 1
          END IF
      IF ((P1 .LT. ANG   .AND.   P2 .GE. ANG) .OR.
     &    (P1 .GE. ANG   .AND.   P2 .LT. ANG))        L=L+1
  1   P1    = P2
      RETURN
      END
C
C
C
      SUBROUTINE REGION (BETA,IREGION)
C     ****************************************************************
C     *        This subroutine will determine which region is        *
C     *        currently being addressed By the program              *
C     ****************************************************************
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON /CONST3/ Z,ZREF,S
      COMMON /REGION/ RS1,RS2,RZ1,RZ2,RO1,RO2
      COMMON /AFB2IN/ HGHT,DELBETA
      COMPLEX*16 BETA
      B         = DREAL(BETA)-DELBETA/5.D0
  IF (Z .GT. S) THEN
                                        IREGION = 1
IF ((B .GT. RO1) .AND. (B .LE. RO2)) IREGION = 6
IF ((B .GT. RS1) .AND. (B .LE. RS2)) IREGION = 3
IF ((B .GT. RZ1) .AND. (B .LE. RZ2)) IREGION = 4
  ELSE
                                        IREGION = 2
IF ((B .GT. RO1) .AND. (B .LE. RO2)) IREGION = 7
IF ((B .GT. RZ1) .AND. (B .LE. RZ2)) IREGION = 8
IF ((B .GT. RS1) .AND. (B .LE. RS2)) IREGION = 5
  END IF
RETURN
END
```